BLITZEINSTIEG

Kickstart/Workbench 2.0/2.1

Autor: Peter Eberlein

media©

Hammerbühlstraße 2
D-8999 Scheidegg

Österreich: INTERCOMP
Heldendankstraße 24 • A-6900 Bregenz

Vorwort

Der Amiga hat inzwischen schon einige Jahre hinter sich gelassen und sein Image vom Rechner für den typischen Freak mehr und mehr verloren. Doch mit seinem neuen Betriebssystem 2.0 stößt er geradewegs in den professionellen Bereich vor. Dies ist für den Anwender natürlich ein großer Vorteil, denn die Programme werden nicht nur immer besser und absturzsicherer, sondern warten auch mit immer mehr Funktionen auf, allen voran die Benutzeroberfläche selbst. Doch damit kommt auch das große Übel professioneller Software auf uns zu: Handbücher, die nicht mehr in Seitenzahlen, sondern in Kilogramm gemessen werden.

Deshalb dieses Buch. Wer die Originaldokumentation zu AmigaOS 2.0 einmal gesehen hat, wird entweder schreiend das Zimmer verlassen haben oder ist in Resignation verfallen. Ausführliche Anleitungen schön und gut, jeder freut sich, wenn all seine Fragen beantwortet werden und nichts tut mehr Not, als ein Nachschlagewerk um dem neuen Betriebssystem Herr zu werden. Doch niemand, der ein wenig Ahnung von Computern hat, wird dieses Machtwerk von vorne bis hinten durchlesen. Und wenn Sie mittendrin aufhören, wer weiß, was Ihnen alles entgeht.

Mit dem Blitzeinstieg halten sie den sprichwörtlichen Rettungsring in Händen. Nach dem Motto „Auspacken und Loslegen" soll Ihnen der Einstieg in die neue Dimension von Amiga-Software so leicht wie nur möglich gemacht werden. Selbstverständlich kann und soll dieses Büchlein die Originaldokumentation des neuen Betriebssystems keinesfalls ersetzen. Doch diese ist für den totalen Neueinsteiger geschaffen, der nicht

weiß, wo sich die Return-Taste befindet. Wer derartige Informationen benötigt, ist mit dem „Handbuch zur Systemsoftware" richtig bedient. Sollten Sie hingegen schon mit einer der älteren Versionen von AmigaOS vertraut sein, oder zumindest mit anderen Computern bereits in Berührung gekommen sein, so ist der „Blitzeinstieg" die bessere Wahl.

Daraus folgen einige Besonderheiten dieses Buch betreffend, die noch schnell geklärt werden sollten, bevor es dann wirklich „blitzartig" losgehen kann. Es wird grundsätzlich auf jegliche Erläuterungen zur Bedienung von Computern allgemein verzichtet, d.h. Sie werden weder erfahren, wie man Disketten schreibschützt noch wie man einen Reset auslöst. Wegen des geringen Umfangs dieses Werks und des dagegen enormen Umfangs des Themas müssen Sie eben gewisse Voraussetzungen mitbringen. Außerdem wird auf die meisten Funktionen, die bereits unter Kickstart 1.3 existierten, nur kurz eingegangen! Dafür ist dieses Büchlein aber kompakt genug um es in einem Stück durchzuarbeiten und wenn Sie anschließend den vollen Überblick haben, können Sie, falls noch Unklarheiten zu einzelnen Themen bestehen, ja immer noch im dicken Handbuch nachblättern. Im übrigen kann der Umgang mit einem Betriebssystem eh nicht gelehrt werden - nur geübt.

Zum Aufbau des Blitzeinstiegs ist noch zu sagen, daß er in zwei Teile untergliedert ist. Ein erster Teil, der sich in erster Linie an den Anwender richtet und sich um die Bedienung von Workbench und Shell dreht. Sie sollten Ihn von vorne durchlesen und nicht nur Sie gerade jetzt interessierende Themen nachblättern, da oftmals Grundlegendes im Verlauf der Besprechung eines Tools erläutert wird, so daß Sie sich

dieses Wissen praktisch nebenbei an Hand eines Beispiels erarbeiten. Und dann ein zweiter, etwas kürzerer Teil, der sich an den weitergehend Interessierten wendet, indem er die Programmierung von AmigaOS 2.0 anspricht. Falls Ihnen nun Zweifel an dem Sinn eines Programmierteils in einem Blitzeinstieg zum Betriebssystem kommen, seien Sie versichert: auch Sie werden darauf zurückgreifen. Denn der Amiga ist schon immer in ganz besonderem Maße von seinen Anwendern programmiert worden (was sicher auch aus den Zeiten als „Freak"-Computer erklärt werden kann) und gerade das neue Betriebssystem lockt mit einer Fülle phantastischer Funktionen, die das Schaffen eigener Programme enorm erleichtern. Und selbst wenn Sie nicht selbst Hand anlegen wollen: Das Verständnis von den internen Vorgängen einer Maschine hat noch niemandem geschadet.
Nachdem ich Sie mit soviel Vorwort hoffentlich nicht gleich verschreckt habe, wünsche ich Ihnen den blitzschnellen Durchblick, und daß auch Sie AmigaOS 2.0 voll ausschöpfen können.

Peter Eberlein

Teil I: Anwendung von AmigaOS 2.0

1. Installation

1.1 Hardware

Dieser erste Teil ist weniger eine Anleitung, als ein Ratgeber für diejenigen, die Ihren Rechner nicht bereits mit installierter Kick 2.0 gekauft haben und sich vielleicht noch fragen, auf welche Weise sie die Nachrüstung durchführen sollen. Dazu gibt es grundsätzlich drei Möglichkeiten:

Sie kaufen einen Kick 2.0 ROM Chip, den sie (gemäß der mitgelieferten Einbauanleitung) gegen Ihren alten Kick 1.2/1.3 ROM Chip auswechseln. Der Vorteil: Diese Methode ist relativ preisgünstig, einfach vorzunehmen und man wird hinterher keinen Unterschied zu einem Amiga sehen, der bereits mit Kick 2.0 ausgeliefert wurde.

Wenn Sie für alle Fälle immer noch auf die alte Kickstart zurückschalten können möchten, so besorgen Sie sich eine Kickstart-Umschaltplatine, die Platz für den alten ROM Chip und den neuen ROM Chip oder einen Satz EPROMs bietet. Achten Sie beim Kauf darauf, daß die Platine auch für Kick 2.0 geeignet ist, da es auch spezielle Umschaltplatinen für 1.2/1.3 gibt. Kick 2.0 im ROM ist die problemlosere Variante, da Sie den Chip überall kaufen können. Die Alternative ist die EPROM-Variante. EPROMs sind Speicherbausteine, die beschrieben werden können und ihre Daten dann bis zur erneuten Löschung behalten, selbst wenn sie von der Stromzufuhr abgetrennt werden. Der Vorteil ihrer Verwendung liegt darin, daß Sie auch neue Varianten des

Betriebssystems auf die gleichen Chips speichern können, und keinen neuen ROM Chip mehr kaufen müssen. Der Nachteil ist jedoch darin zu sehen, daß Sie einen EPROM-Brenner benötigen (mit dem die Dinger beschrieben werden). Ein nicht ganz billiger Spaß, und eher dem Fortgeschritteneren anzuraten, weniger dem Einsteiger. Und schließlich noch die einfachste und billigste Möglichkeit: gar keine Hardwareeingriff vornehmen, sondern die neue Kickstart einfach von Diskette (oder Festplatte) laden. Warum nicht gleich, wenn es auch so geht? Nun, um so vorgehen zu können, ist eine MMU vonöten, die sich normalerweise nur dann in Ihrem Rechner befindet, wenn Sie über eine Turbokarte verfügen. Sollten Sie jedoch damit ausgerüstet sein, sollten Sie sich des Programms ZKick bedienen, daß auf PD-Basis verfügbar ist. Bedenken Sie jedoch auch, daß beim Laden ins RAM ein halbes Mega Byte nutzbaren Speichers verloren geht, denn irgendwo muß die Kickstart ja hin! Man kann also sagen, daß diese Methode mehr eine Übergangslösung darstellt, auf die Dauer werden Sie wohl um einen kleinen Hardwareeingriff nicht herumkommen.

Wenn Sie erst einmal die Kickstart im Rechner haben, können Sie gleich voll loslegen. Es sei jedoch auch darauf hingewiesen, daß alle Funktionen des neuen Betriebssystems nur dann genützt werden können, wenn die entsprechenden Custom-Chips vorhanden sind. Dabei handelt es sich um den neuen Agnus, der neben vielen versteckten Verbesserungen vor allem mehr Chip Memory zuläßt, das beispielsweise für graphische Anwendungen besonders dringend benötigt wird, und die ECS-Denise, mit der es dann auch möglich wird, die neuen Grafikmodi zu benützen. Beide Chips sind mit entsprechenden Einbauanleitungen versehen und wie gesagt nicht unbedingt erforderlich, so daß an dieser Stelle nicht weiter darauf eingegangen werden braucht. Angeraten seien sie Ihnen aber auf alle Fälle.

Noch ein kleiner Tip am Rande: Zu dem Zeitpunkt da dieses Buch entsteht existieren drei gültige Kickstart 2.0 Versionen: Die 37.175, die eigentlich „offizielle", die auch üblicherweise verkauft wird, die 37.210, die ein paar weniger Fehler enthält, aber noch nicht auf ROMs erhältlich ist und die 37.300, die im A 600 eingebaut ist. Diese und sehr wahrscheinlich bald noch andere Versionen sollten Sie jedoch nicht beunruhigen, denn abgesehen von ein paar kleinen Bug-Fixes und, im Falle der 37.300, dem Card.device, das speziell für den A 600 nötig ist, gibt es keine wesentlichen Unterschiede.

1.2 Software

Bei der Software ist es schon ein wenig schwieriger. Wenngleich ähnlich wie die ROM Version 37.175 auch eine offizielle Workbench Version existiert (37.67), so ist bereits jetzt abzusehen, daß in Kürze eine neue Workbench auf den Markt kommen wird. Die Entwicklerversion läuft zur Zeit unter der Bezeichnung Workbench 2.1, was sich jedoch bis zum offiziellen Verkauf noch ändern kann. Da sie jedoch einige gewaltige Neuerungen enthält und dieses Buch natürlich so aktuell wie möglich sein soll, wird an geeigneten Stellen auch auf diese Version eingegangen werden, selbst wenn sich durchaus noch einiges bis zum endgültigen Erscheinen ändern kann. Doch bis der Blitzeinstieg in den Regalen steht, ist die neue Workbench vielleicht schon erhältlich. Und sie enthält wirklich bemerkenswerte Neuerungen. Doch dazu später mehr.

Nun geht es erst einmal darum, die Software auf Ihrem Rechner zu installieren. Sollten Sie nicht über eine Festplatte verfügen, so ist es unbedingt notwendig, erst einmal Sicherheitskopien von den Disketten zu machen. Dies kann sowohl mit dem ganz

normalen Diskcopy von der Workbench aus geschehen (Workbench laden, Diskette anklicken und dann aus dem Icons Menü „Copy" anwählen) oder mit einem der professionellen Kopierprogramme, beispielsweise Tetra Copy. Recht viel mehr können Sie dann im Moment nicht tun, denn die Veränderung der Diskетteninhalte gehört dann bereits zur Konfiguration, die im nächsten Kapitel behandelt wird. Dort wird dann kein Unterschied mehr zu den Festplattenbesitzern gemacht, da im Grunde alles gleich abläuft, Sie nur des öfteren irgendwelche Disketten wechseln werden müssen.

Zur Installation auf eine Festplatte bedienen Sie sich der Installationsdiskette. Von ihr wird gebootet, so daß Sie nach kurzer Zeit bereits einen Workbench 2.x Screen vor sich haben. Dort öffnen Sie das Disketten Icon der Installationsdiskette und sehen sich daraufhin den unterschiedlichen Installationsprogrammen gegenüber. Bei WB 2.0 sind dies im wesentlichen „PrepHD", „FormatHD", InstallHD" und „UpdateWB", bei WB 2.1 sind jeweils mehrere Versionen der entsprechenden Installationsprogramme in der Ordern „HDSetup", „Install 2.1", „Install Printers" und „Install Languages". Die unterschiedlichen Versionen und der „Install Languages" offenbaren dann auch eine der beiden umwerfenden Neuerungen von WB 2.1, sie ist „localized", d.h. Sie können die Workbench in Ihrer Sprache, also auch deutsch, installieren. Und zukünftige Programme, die dieses neue Feature nützen, können Ihnen sofort verständlich entgegentreten! Doch nun geht es erst einmal darum, alle notwendigen Daten auf die Platte zu werfen. Gehen wir davon aus, daß diese bereits soweit präpariert ist, daß sie als DH0: oder ähnlichem zur Verfügung steht. Die notwendigen Formatierungsaufgaben werden ja üblicherweise mit dem Host-Adapter spezifischen Programm gemacht. Sollte dies nicht der Fall sein, und sie über einen A 2091 oder einen A 3000 mit nicht

installierter Festplatte (?) verfügen, so starten Sie „PrepHD", „FormatHD" und „InstallHD", bzw. eins der Programme aus dem „HDSetup" Schub der WB 2.1. Die Bedienung ist menügesteuert und sollte keine Probleme bereiten.

Ähnlich einfach ist die Installation der Workbench selbst. Nach dem Doppelklick auf „UpdateHD" werden Sie nach dem Namen Ihrer Festplattenpartition gefragt, die Sie mit WB 2.0 beglücken wollen, und nachdem Sie wahrscheinlich „DH0:" oder ähnliches eingegeben haben, können Sie noch die bevorzugte Tastaturbelegung wählen („d" für deutsch, wählen Sie folglich „4") und alles andere läuft automatisch. Wenn Sie zu den mißtrauischeren Amiga-Besitzern gehören, und gerne wüßten, was da eigentlich vorgeht und ob das auch alles so abläuft, wie Sie das gerne hätten: Recht viel anderes bleibt Ihnen nicht übrig. Denn die Installation von Hand durchzuführen, ist äußerst aufwendig. Viel sinnvoller ist es, das Installationsprogramm erst mal seinen Job machen zu lassen, und anschließend wieder aufzuräumen. Wenn Sie bevorzugt mit der Workbench arbeiten, wird dazu sowieso kein Anlaß bestehen, lediglich Shell-Benutzer bevorzugen ihre eigene „Startup-Sequence" und alle Befehle im „C:" Directory, und nicht im „System" Ordner. Wie dem auch sei, auf jeden Fall werden Ihre alte „Startup-Sequence" und andere, wichtige Konfigurationsdateien in ein neues Verzeichnis namens „old" kopiert, so daß eigentlich nur wenig schief gehen kann.

Bei der Workbench 2.1 läuft das ganze ein wenig anders. Hier werden Sie bereits mit dem neuen Installationsprogramm konfrontiert, daß in Zukunft auch von anderen Anwendungen genutzt werden wird. Der Vorteil liegt darin, daß es Ihnen erlaubt, je nach Ihrem Wissensstand die Installation mehr oder weniger zu automatisieren. Mit

der Wahl des „Experten" können Sie praktisch jeden Vorgang vor der Ausführung nochmals bestätigen lassen. Außerdem können Sie zu jeder Frage, die Ihnen gestellt wird, einen Hilfstext abrufen, der Ihnen klar macht, was Ihre Entscheidung eigentlich bewirkt. Wenn sich dieses System erst einmal durchgesetzt hat, werden Sie wohl keinen „Blitzeinstieg" mehr brauchen...

2. Die neue Workbench

Bevor wir uns nun in all die Programme vertiefen, die zur neuen Workbench gehören, ist vielleicht eine kleine Einführung die Workbench selbst betreffend ganz angebracht. Denn wenngleich dem geübten Amiga Anwender sicherlich gleich die neuen Gadgets ins Auge springen, und ihn niemand davon abhalten kann, sie sofort auszuprobieren, so kann eine kurze Übersicht doch nicht schaden. Und vielleicht kommen Sie so sogar schneller ans Ziel.

2.1 Boot Menu

Vielleicht nicht das erste, was Ihnen auffällt, aber doch das erste, was nach dem Einschalten Neues kommt, ist das „Boot Menu". Natürlich ist auch die Workbench-Hand verschwunden und gegen eine kleine Animation mit einer Diskette ersetzt worden (die Sie, sollten Sie eine Festplatte besitzen, ohnehin nie zu Gesicht bekommen). Doch wenn Sie direkt nach dem Reset beide Maustasten gleichzeitig gedrückt haben, gelangen Sie in das Bootmenü. Hier werden alle zur Verfügung stehenden Boot Partitionen angezeigt, und mittels eines Klicks auf das entsprechende Gadget booten Sie von einer Partition Ihrer Wahl. Hier sind zunächst einmal alle ange-

schlossenen Diskettenlaufwerke angezeigt, dann alle Partitionen der Festplatte(n) und auch resetfeste RAM-Disks, wie die RAD: oder die VD0:. Dies ist eine äußerst sinnvolle Einrichtung, denn so ist es beispielsweise möglich, von einem externen Diskettenlaufwerk zu booten, auch wenn die Festplatte mit ihrer Priorität über diesem liegt. Die Prioritäten und einige andere Informationen können Sie übrigens mittels der „Advanced Options..." abrufen.

In diesem Menü können auch Partitionen ganz ausgeschaltet werden, wenn z.B. ein Spiel die Festplatte nicht verträgt, brauchen Sie nicht an irgendwelchen Schaltern zu hantieren, sondern Sie melden die Festplatten-Partitionen einfach ab (auf das „Enabled"-Gadget klicken). An weiteren Informationen ist der DOS-Type angezeigt: „0" für das (alte) OldFileSystem (OFS) und „1" für das neue FastFileSystem (FFS). Dann die schon angesprochene Bootpriorität, die über die Reihenfolge beim Bootvorgang entscheidet. Im Normalfall wird erst einmal im internen Laufwerk nachgesehen, ob eine bootbare Diskette vorhanden ist, und wenn nicht, bei der Festplatte usw. Negative Bootprioritäten bedeuten, daß von diesen Partitionen nie gebootet wird. Der „Device"-Name und die „Unit"-Nummer sind auch noch angezeigt, aber weniger interessant. Schließlich läßt sich sogar die „Startup-Sequence" ausschalten, so selten Sie diese Funktion auch gebrauchen können werden.

2.2 Gadgets

Was nun aber wirklich ins Auge springt, ist das neue Outfit, alles plastisch, alles in dezentem grau, alles irgendwie professionell - so soll es zumindest aussehen. Und in der Tat wurden einige recht attraktive Veränderungen durchgeführt. So fällt zunächst

einmal auf, daß am rechten oberen Eck nur noch ein Vorder-/Hintergrund-Gadget angebracht ist und dafür ein neues Gadget, das „Zoom"-Gadget hinzugekommen ist. Die Bedienung des neuen Vorder-/Hintergrund-Gadgets ist ein wenig gewöhnungsbedürftig, aber gar nicht so schlecht. Sie können jetzt zwar nicht mehr mit einem einzigen Klick, das entsprechende Fenster nach vorne bzw. nach hinten legen, so wie es Ihnen gefällt. Von nun an entscheidet das System selbst, was für Sie das beste ist, doch dafür läßt sich schneller durch viele Windows klicken, die alle die gleiche Anordnung haben, da Sie die Maus nicht mehr neu positionieren müssen. Doch wie funktionieren diese Dinger nun? Also, wenn Sie das Gadget anwählen und das zugehörige Fenster ist von irgendeinem anderen Fenster zumindest teilweise verdeckt, dann wird das angewählte in den Vordergrund geholt. Befindet es sich bereits im Vordergrund, so kommt es ganz nach hinten. Wollen Sie also ein teilweise bedecktes Fenster in den Hintergrund verbannen, so ist ein Doppelklick nötig (besser gesagt, zwei Klicks, aber das spielt in diesem Fall keine Rolle).

Das dafür neu gewonnene „Zoom"-Gadget beschleunigt die Größeneinstellung von Fenstern. Meist ist es gar nicht nötig, eine bestimmte Größe einzustellen, sondern man will nur schnell Platz für ein anderes Fenster schaffen. Dazu wählt man das „Zoom"-Gadget an, wodurch das Fenster auf seine minimale Größe verkleinert wird. Anschließend kann durch einen erneuten Klick die ursprüngliche Größe wiederhergestellt werden. Doch es muß nicht immer ein Umschalten zwischen Minimum und Maximum sein. Vielmehr wird die aktuelle Größe bei der Anwahl des „Zoom"-Gadgets gespeichert. Daher ist es folglich kein Problem, die beiden alternativen Einstellungen mit dem normalen Größen-Gadget zu modifizieren.

Die übrigen Gadgets von Fenstern haben sich in ihrer Funktion nicht geändert, lediglich im Aussehen. Es bleibt beim Schließ-Symbol links oben, dem Rollbalken rechts und dem Gadget zur Einstellung der Größe rechts unten. Jedoch gibt es auch ganz neue Gadget-Typen, die allerdings in den folgenden Kapiteln bei den Voreinstellern vorgestellt werden, damit Sie sofort einen Bezug zur Anwendung und auch gleich ein Beispiel geliefert bekommen.

2.3 Menüs

Die Workbench-Menüs wurden auch noch einmal kräftig aufgepeppt, so daß sie sowohl mehr Funktionen bieten, als auch logischer gegliedert sind. Vor der genaueren Betrachtung noch eine kleine Anmerkung: Viele Menüpunkte sind in „ghostwriting" dargestellt, also in Geisterschrift, oder etwas konkreter ausgedrückt, nicht richtig lesbar. Dies weist meist darauf hin, daß Sie ein oder mehrere Icons auswählen müssen, bevor die Funktion anwählbar ist. Dies geschieht üblicherweise mittels eines einzigen Mausklicks, doch wenn Sie mehr als ein Icon anwählen möchten, müssen Sie gleichzeitig die Shift-Taste gedrückt halten. Oder aber Sie bedienen sich der neuen Selektionsmöglichkeit der Workbench 2.0: Einfach irgendwo ins Fenster klicken, wo gerade kein Icon oder sonst irgend etwas im Weg ist, und anschließend die Maus (bei gedrückter Maustaste) ziehen. Dann erscheint ein rechteckiger, pulsierender Kasten und alle Icons, die sich in ihm befinden, werden ausgewählt, sobald Sie den Mausknopf wieder loslassen. Da aber nur rechteckige Ausschnitte möglich sind, wird der Weg zur Tastatur leider doch nicht immer vermeidbar sein.

2.3.1 Workbench

Gleich im ersten, dem „Workbench"-Menü wartet ein neuer Menüpunkt: „Backdrop". Denn die Workbench kann nun sowohl als Screen als auch als Window auftreten. Wenn Sie die bekannte Variante (den Screen) bevorzugen, so wählen Sie „Backdrop" an, hätten Sie auch die Workbench lieber in einem Fenster, so muß dieser Menüpunkt abgewählt werden, d.h. es darf kein Haken neben dem Text stehen. Der Vorteil der Window-Variante ist natürlich, daß alle Disk-Icons immer leicht zugänglich sind, da Sie ja das Window problemlos in den Vordergrund holen können. Bei einem Screen müßten alle Fenster zur Seite geschoben werden. Um Ihre Wahl dauerhaft zu speichern sollten Sie noch im „Window"-Menü den Menüpunkt „Snapshot" aufrufen.

Bei Anwahl von „Execute Command..." erscheint ein kleines Fenster, in dem Sie den Namen eines Programms eintippen können, das Sie gerne ausführen möchten. Dadurch braucht auch der reine Workbench-Benutzer nicht extra eine Shell öffnen, wenn er ein Shell-Kommando starten will. Falls eine Ausgabe stattfindet, so wird ein Fenster geöffnet und der entsprechende Text darin angezeigt. Anschließend müssen Sie dieses Fenster dann mit dem Close-Gadget links oben wieder schließen. Weitere Kommandos können nicht unmittelbar ausgeführt werden, aber wie gesagt, für einen einzigen Befehl ist diese Funktion durchaus sinnvoll.

„Redraw All" ist nur dann nötig, wenn ein fehlerhaftes Programm die Workbench-Grafik verunstaltet hat. Denn diese Funktion veranlaßt die Workbench, den Bildschirm nochmals komplett neu aufzubauen. Wenn das schuldige Programm neben der Grafik

auch noch Programmspeicher überschrieben haben sollte, werden natürlich nicht nur alle Fenster neu aufgebaut, sondern auch die zugehörigen Verzeichnisse erneut gelesen. Denn wenn Sie beispielsweise ein Programm und sein Icon über die Shell aus einem Verzeichnis löschen, dessen Window gerade auf der Workbench angezeigt wird, so erfährt die Workbench davon nichts. Mit anderen Worten: Sie sehen möglicherweise Dinge angezeigt, die überhaupt nicht existieren. „Update All" behebt das Problem indem die Daten aktualisiert werden (nicht nur ihre graphische Darstellung). Wenn Sie natürlich immer auf der Workbench bleiben und auch Ihre Anwenderprogramme die Workbench nicht hinterrücks überrumpeln, werden Sie diese Funktion nicht benötigen, da dann ja sowieso alles seine Ordnung hat.
„Last Message" kennt man ja schon von der WB 1.3, es dient der erneuten Anzeige einer evtl. aufgetretenen Fehlermeldung in der Titelzeile, was immer dann sinnvoll ist, wenn man diese Meldung nicht mitbekommen hat, weil z. B. ein Fenster die Anzeige verdeckte.

Auch „About" darf natürlich nicht fehlen, damit Sie sich immer vergewissern können, welche Kickstart- und Workbench-Version Sie gerade benützen.
Und schließlich noch „Quit" - der Menüpunkt, der die Workbench über die Klinge springen läßt (natürlich nach vorheriger Bestätigung). Seien Sie jedoch gewarnt! Wenn Sie kein Shell- bzw. CLI-Fenster geöffnet haben und auch kein Anwenderprogramm, das die Möglichkeit bietet, andere Programme zu starten, dann haben Sie den Ast auf dem Sie saßen abgesägt. Denn wenn Sie keine Programme mehr starten können, bleibt Ihnen nicht viel mehr übrig, als den Rechner durch einen Neustart wieder benutzbar zu machen.

2.3.2 Window

Das, was „New Drawer" im „Window"-Menü leistet, hat der geplagte Workbench-Anwender schon lange vermißt: die Möglichkeit, eine neue Schublade zu erzeugen. Bisher mußte der standardmäßig vorhandene „Empty" Drawer erst kopiert und anschließend umbenannt werden. Das hat jetzt ein Ende, Sie wählen einfach das Window aus, in dem Sie gerne das neue Unterverzeichnis erstellt haben möchten und gehen auf „New Drawer". Es erscheint sofort eine neue Schublade mit dem Namen „Unnamed1" und gleichzeitig ein Fenster, in dem Sie diesen Namen ändern können. Das ist die logische Vorgehensweise, die Sie wahrscheinlich schon immer einschlagen wollten.

Ähnlich anwenderfreundlich ist „Open Parent". Wenn Sie sich einige Stufen tief in die Verzeichnisstruktur bewegt und alle vorhergehenden Windows wieder geschlossen haben um Platz zu schaffen, und nun doch noch etwas aus dem übergeordneten Verzeichnis benötigen, so wählen Sie einfach diese Funktion. Sie müssen nicht mehr wie bisher alle Verzeichnisse ein zweites Mal öffnen, sondern eben nur das, das Sie eben öffnen wollten. Auf diese Weise können Sie sich natürlich auch Schritt für Schritt wieder nach oben arbeiten, wenn Sie so schneller zum Ziel kommen.

Bereits bekannt und immer noch überflüssig ist „Close", da Sie Windows weitaus schneller mit einem Doppelklick schließen können, als erst „Close" aus dem Menü anzuwählen. Lediglich wenn das Fenster so sehr überlagert ist, daß Sie zwar irgendwo ein kleines Eck davon erspähen, nicht aber das Close-Gadget, kann die alternative Vorgehensweise eher zum Ziel führen. Jedoch laufen Sie in einem solchen Fall auch

Gefahr, das falsche Window zu schließen. „Update" ist von der Funktionsweise der von „Update All" aus dem „Workbench"-Menü entsprechend, es werden jedoch nicht alle Daten erneut eingelesen sondern nur die aus dem angewählten Fenster.
„Select Contents" wählt alle Icons aus dem aktiven Fenster aus und erspart es Ihnen, mittels der erweiterten Auswahl jedes Icon von Hand anzuklicken.
„Clean Up" ist die ideale Funktion für Leute, die es eilig haben: Sie räumt den Inhalt des gewählten Fensters aus, indem die Icons in einem sauberen Raster plaziert werden, und zwar abhängig von der Größe dieser Icons. Das Herumschieben von Hand entfällt. Zwar nichts Neues, aber immer noch unvermindert brauchbar.
Mit „Snapshot" könne Sie einige Informationen bezüglich des aktiven Fensters speichern. Das Untermenü „Window" fixiert die Größe, Position und die „Show"- und „View by"-Modes (siehe unten). Mit dem Untermenü „All" speichern Sie zudem die Positionen aller Icons, die Sie vorher entweder mit „Clean Up" oder von Hand angeordnet haben können. Wenn Sie das Fenster das nächste Mal öffnen, wird alles wieder auf seinem Platz sein.

Wieder eine neue Funktion ist „Show". Da üblicherweise nicht alle Programme mit Icons ausgestattet sind, Sie aber vielleicht auf diese Zugriff haben möchten, ohne vorher ins Shell zu müssen, können Sie durch Anwahl von „All Files" auch diesen Icons zuordnen. Dabei wird selbständig entschieden, ob es sich um ein Verzeichnis oder ein File handelt und das zugehörige Symbol angezeigt. Die Programme können dann ganz einfach mittels Doppelklick gestartet werden, was genau den gleichen Effekt hat, wie wenn Sie auf „Execute Command" gegangen wären und den Pfad- und Filenamen eingegeben hätten. Um zur normalen Darstellung zurückzukommen, können Sie dann

„Only Icons" selektieren, woraufhin wieder nur die tatsächlich vorhandenen Icons angezeigt werden.
Mit „View By" näheren Sie sich dem Shell sogar noch einen weiteren Schritt. Neben „Icons", der Darstellung, die Sie gewöhnt sind, kann der Inhalt eines Verzeichnisses nun auch in Textform angezeigt werden, so wie man es vom „List"-Befehl im Shell kennt. Dabei stehen Ihnen drei Sortierkriterien zur Auswahl: „Name" (alphabetisch), „Date" (nach Datum) und „Size" (nach Größe). In Verbindung mit „Show" bekommen Sie schon fast alle wirklich vorhandenen Dateien angezeigt. „Fast", weil immer noch die „.Info"-Files herausgefiltert werden, die ja die Icon-Daten enthalten. Doch von der Workbench aus sind diese ja nicht weiter von Interesse.

2.3.3 Icons

„Open" ist der erste und nicht weiter überraschende Menüpunkt: Mit ihm öffnen Sie ein Fenster bzw. starten ein Programm. Natürlich geht das auch mit einem Doppelklick, und noch dazu viel schneller.
„Copy" ist eine interessante Funktion, die mehr in sich hat, als man es von der WB 1.3 gewöhnt ist. Prinzipiell stehen Ihnen ja zwei Möglichkeiten zur Verfügung, Dateien, Verzeichnisse oder Disketten zu kopieren: Sie ziehen das entsprechende Icon in das gewünschte Fenster bzw. auf das Ziellaufwerk. Oder aber Sie rufen „Copy" auf. Der Unterschied besteht darin, daß Sie mit der ersten Funktion immer von einem Verzeichnis in ein anderes kopieren, bei der zweiten aber eine Kopie im gleichen Verzeichnis erstellen, die dann „Copy_of_" und dem jeweiligen Namen genannt wird. Dies ist immer dann sinnvoll, wenn Sie eine Sicherheitskopie auf der gleichen Einheit erstellen wollen, oder beispielsweise eine Datei verändern möchten, aber dennoch

das Original behalten. Dann können Sie später immer wieder auf die „Copy_of_" Datei zugreifen. Doch halt, was war das mit den Disketten? Ja, genau, bei der WB 2.0 gibt es keinen „Diskcopy"-Befehl mehr, diese Funktion wird ebenfalls von „Copy" übernommen. Dazu müssen Sie lediglich das Disketten-Icon anwählen und anschließend „Copy". Daraufhin wird diese Diskette über das Laufwerk, in dem sie sich befand, kopiert und Sie regelmäßig zum Wechseln von Quell- und Zieldiskette aufgefordert. Wenn Sie über zwei Laufwerke verfügen können Sie natürlich auch direkt von dem einen auf das andere kopieren. Hierzu ziehen Sie das eine Disketten-Icon auf das andere.

„Rename" tut genau das, was der Begriff aussagt: Dateien umbenennen. Die Handhabung ist denkbar einfach: Icon anklicken, „Rename" wählen und im Text-Gadget die Bezeichnung ändern.

Um Programmen Parameter zu übergeben, benötigen Sie „Information". Wird dieser Menüpunkt bei angewähltem Icon aufgerufen, so erscheint ein Fenster, in dem nähere Informationen zu dem zugehörigen Programm enthalten sind. Da wird das Icon nochmals angezeigt, die Größe, die Anzahl der Blöcke, die es auf einem Datenträger benötigt und wann die letzte Änderung vorgenommen wurde. Darüber hinaus sind aber auch Felder vorhanden, die Sie editieren können.
Da wäre zunächst einmal das „Stack"-Feld, in dem die Größe des Stapelspeichers eingegeben werden kann. Im Normalfall sollten Sie diesen Wert nicht ändern, nur bei wenigen Programmen, die ausdrücklich darauf hinweisen, kann es unter Umständen nötig werden, den Wert zu erhöhen. Unter 4096 sollte er jedoch niemals gewählt werden. Weiterhin können Sie dem File noch ein „Comment" mitgeben. Diese

Information läßt sich dann auch von der Shell aus mittels „filenote" abfragen bzw. ändern.
Und um auch den „protect"-Befehl dem Workbench-Anwender zugänglich zu machen, sind rechts oben sechs Gadgets, die den Status der Flags bestimmen. Neben den bekannten („Readable" = lesbar, „Writable" = schreibbar, „Executable" = ausführbar und „Deletable" = löschbar) sind noch zwei für AmigaDOS neue Flags hinzugekommen: „Script" bedeutet, daß ein File eine Script-Datei ist, woraufhin diese nicht mehr mit „execute" gestartet werden muß sondern sich wie jedes andere Programm verhält. „Archived" schließlich ist ein Flag, daß bei jedem schreibendem Zugriff auf die Datei automatisch gelöscht wird. Dadurch ist es leicht möglich festzustellen, ob sich eine Datei verändert hat. Dies ist besonders für Backup Programme sehr hilfreich, da dann nur die veränderten Dateien gesichert werden müssen.

Doch die wichtigste und auch schon angesprochene Funktion ist die Parameterübergabe beim Aufruf von Programmen. Dazu gibt es das „Tool Types"-Feld, in dem Sie die gewünschten Parameter eingeben können. Das Format ist üblicherweise „Schlüsselwort = Argument". Welche Schlüsselworte Ihnen zur Verfügung stehen, ist bei der Workbench 2.0 bereits in dem „Tool Types"-Feld angegeben, und zwar in spitzen Klammern. Das bedeutet, daß diese Angaben überlesen werden. Bei älteren Programmen ist dies oftmals nicht der Fall, dann sind die Parameter aber in der Dokumentation zu dem Programm erklärt.

Wollen Sie nun einen neuen Parameter eingeben, so klicken Sie „New" an und können anschließend in dem Textfeld, in dem sich nun der Cursor befindet, die Parameterzeile nach obigem Format eingeben. Um eine bereits bestehende Zeile zu verändern,

genügt es, diese anzuklicken, woraufhin Sie diese dann im unteren Textfeld editieren können. Sie können auch Zeilen wieder löschen, indem Sie diese zuerst anwählen und dann auf „Del" klicken. Jegliche Änderung sollten Sie nach der Bearbeitung durch Klick auf das „Save"-Gadget abspeichern.

Die Funktion „Snapshot" macht für einzelne Icons genau das, was die „Snapshot"-Funktion im „Window"-Menü mit einem ganzen Fenster macht: Sie fixiert die Position des Icons. Dadurch erscheint es beim nächsten Öffnen seines Mutterfensters an der vorher bestimmten Stelle. Im allgemeinen wird man zwar alle Icons so positionieren, wie man es sich vorstellt und dann die Einstellungen in einem Rutsch abspeichern, doch gerade im Diskettenbetrieb kann das doch etwas dauern. Wenn Sie also nur ein neu hinzugekommenes Icon fixieren wollen, so ist diese Funktion effektiver.
„Unsnapshot" hebt diese Einstellungen wieder auf, woraufhin die Workbench die Icons wieder frei positionieren kann. Dies ist besonders dann sehr hilfreich, wenn Sie in einer Schublade sehr oft Dateien legen und andere wieder entfernen. Denn in diesem Fall ist es einfacher, es der Workbench zu überlassen, die Icons halbwegs sinnvoll zu plazieren, als jedesmal dem neuen Zustand entsprechend alle Positionen erneut zu fixieren.

Eine weitere Verbesserung der Effektivität bringt „Leave Out". Da häufig benötigte Programme grundsätzlich am tiefsten in der Verzeichnisstruktur zu finden sind (Murphy's Gesetz), müssen Sie sich vor deren Start immer erst durch eine Unzahl von Windows wühlen. Viel einfacher wäre es doch, das Icon direkt auf der Workbench zu haben. Und genau dazu ist „Leave Out" da. Das Angewählte Icon wird auf die

Workbench gelegt, wenngleich sich die Position des Files selbst nicht verändert. Aber Sie können immer sofort auf das Programm zugreifen, selbst nach einem Neustart. Wenn Sie das Icon wieder an seien eigentlichen Platz zurücklegen möchten, bedienen Sie sich des Menüpunkts „Put Away". Das Icon verschwindet von der Workbench und taucht in seinem ursprünglichen Window wieder auf.

„Delete" ist eine jener Funktionen, die wieder einmal genau das tut, was man von ihr erwartet: Es wird das angewählte Programm gelöscht! Zwar erst nach Rückfrage, dann aber unwiderruflich. Eine weniger grausame Methode (grausam für Sie, wenn Sie die Datei später doch noch brauchen) stellt ja der bekannte „Trashcan" dar. Doch werden die Files dabei ja auch nicht gelöscht, sondern nur umgelagert. Wenn Sie wirklich Platz auf Ihrem Datenträger brauchen, dann müssen Sie entweder den „Trashcan" leeren (s.u.) oder gleich „Delete" anwenden.

Der nächste Menüpunkt, „Format Disk..." ist nur anwählbar, wenn Sie vorher auf ein Diskettensymbol geklickt haben. Anschließend ist es möglich, diese Diskette zu formatieren, d.h. komplett zu löschen. Bei einer neuen Diskette, oder einer solchen, die in einem fremden Format bespielt war, müssen Sie bei der folgenden Sicherheitsabfrage „OK" anwählen. Nur wenn die Diskette bereits im normalen AmigaDOS-Format bespielt war, sollten Sie „OK-QUICK" anklicken, da bei dieser Variante lediglich der Root-Block der Diskette gelöscht wird, was zwar sehr viel schneller geht, aber sofort Fehler verursacht, wenn auf einen uninitialisierten Block zugegriffen wird. Die „Quick" Methode löscht natürlich auch nicht wirklich alle Daten, sondern im Normalfall nur deren Namen und den Zeiger auf ihre Position. Das

ermöglicht es einigen schlauen Disketten-Utilities, die Daten wiederherzustellen, während bei der vollständigen Formatierung nichts mehr zu machen ist. Doch verlassen sollten Sie sich darauf auf keinen Fall!
Der letzte Punkt dieses Menüs, „Empty Trash" funktioniert nur im Zusammenhang mit einem angewählten „Trashcan". Mit ihm kann man den „Papierkorb ausleeren", mit anderen Worten, all die Files, die Sie irgendwann man in den „Trashcan" geworfen haben, werden endgültig gelöscht. Damit sind Sie ebenso verloren, wie wenn Sie gleich „Delete" angewählt hätten; nur solange die Files noch im Papierkorb liegen, können Sie problemlos zurückkopiert werden.

2.3.4 Tools

Dieses Menü ist nicht für die Workbench selbst, sondern vielmehr für Anwenderprogramme reserviert. Falls eines Ihrer Programme also diese Funktion unterstützt, so können Sie hier einen neuen Menüpunkt installieren und das Programm dann über diesen starten, anstatt ein Icon anzuklicken. Wie Sie zu diesem Zweck vorzugehen haben, ist von Anwendung zu Anwendung unterschiedlich; ziehen Sie diesbezüglich die Dokumentation des Programms zu Rate.

Einen Punkt weist dieses Menü aber doch schon auf, nämlich „ResetWB". Was man mit ihm anstellt, dürfte wohl intuitiv klar sein: die Workbench zurücksetzen. „Zurücksetzen" bedeutet dabei, daß sämtliche Windows geschlossen, die Workbench in ihren Anfangszustand versetzt und die vorher vorhandenen Windows wieder geöffnet werden. Wann Sie jemals in die Verlegenheit kommen werden, diesen Menüpunkt aufrufen zu müssen, ist unklar. Sollten aber wirklich einmal seltsame Fehler auf der

Workbench auftreten, kann es auf keinen Fall etwas schaden, diese Funktion anzuwählen - wenngleich die Aussichten auf einen Erfolg gering sind. Aber einen Versuch ist es wert!

3. Konfiguration

3.1 Preferences

Nun, da Sie entweder das gesamte Paket auf Ihrer Festplatte oder die einzelnen Disketten als Arbeitskopie vorliegen haben, sollte die Workbench an Ihre Bedürfnisse angepaßt werden. Nur so können Sie die volle Leistung aus dem Rechner holen. Der erste Schritt sind die Preferences, die nicht mehr wie in den Versionen 1.2 und 1.3 in einem Programm vereint sind. Bei AmigaOS 2.0 gibt es für jeden Bereich ein eigenes Preferences Programm und abgesehen von den Workbench Preferences werden beispielsweise auch zu Erweiterungshardware zusätzliche Preferences Programme mitgeliefert, die Sie im „Prefs" Ordner ablegen sollten.

Bei der Besprechung der Preferences werden wir gleich einige Neuerungen in der Bedienung von AmigaOS 2.0 allgemein kennenlernen, und brauchen uns später keine anschaulichen Beispiele mehr ausdenken! Bitte bedenken Sie, daß sämtliche Text,- Gadget- und Menübezeichnungen im Folgenden mit ihrer englischen Originalbezeichnung genannt sind. Sollten Sie bereits über die Workbench 2.1 verfügen und eine andere Sprache angewählt haben, differieren die Anzeigen natürlich.

3.1.1 Time

Auch wenn Sie eine Echtzeituhr besitzen, so muß diese zumindest ein erstes Mal eingestellt werden. Dazu (und zur Einstellung nach jedem Neustart, wenn Sie keine besitzen) dient der Voreinsteller Time. Er eignet sich für den Anfang besonders gut, weil wir ganz genau wissen, was wir nun einstellen wollen - eben die aktuelle Uhrzeit - und er bereits eine Neuerungen enthält, die erläuterungsbedürftig sind.

Starten Sie also das Programm „Time", das sich im Ordner „Prefs" befindet. Gleich oben ist in einem Kästchen ein Monatsname zu lesen, und links davon ein gebogener Pfeil der auf sich selbst zeigt. Dieses neue Symbol zeigt einen Gadget-Typ an, der, jedesmal wenn er angeklickt wird, seinen Inhalt ändert. Dadurch kann man sich im Fall der Monatsnamen durch wiederholtes Klicken bis zu dem Monat vorarbeiten, den man möchte. Ist man bei Dezember angelangt, wird automatisch wieder von vorne, also mit Januar begonnen. Diese neuen Gadgets, die im übrigen „Blättersymbole" genannt werden, werden Ihnen in Zukunft öfter über den Weg laufen, und es ist oftmals sinnvoll, erst einmal alle Möglichkeiten durchzuklicken, damit man überhaupt weiß, was alles zur Auswahl steht. Stellen Sie nun also den richtigen Monatsnamen ein. Dabei verändert sich die darunterliegende Anzeige des Kalenders, um die korrekte Anzahl von Tagen und die Verteilung der Wochentage wiederzugeben. Klicken Sie einfach auf den aktuellen Tag, Neues gibt es hier nicht zu berichten. Ebenso verhält es sich mit der Jahreszahl, die ganz einfach ein Text-Gadget darstellt, in dem sie mittels Tastatur das Jahr eingeben können. Schließlich sind Schieberegler zur Einstellung der Uhrzeit vorhanden.

Mit den drei unteren Gadgets können Sie nun das Ergebnis Ihrer Bemühungen entweder in die Echtzeituhr übernehmen (Save), die Systemuhr setzen, aber nicht die Echtzeituhr verändern (Use) oder ganz verwerfen (Cancel). Diese drei Gadgets sind auch in allen anderen Preferences-Editoren vorhanden, und bedeuten allgemein immer Abspeichern, Benutzen oder Verwerfen. Doch verlassen wir nun den Time-Editor und wenden uns dem nächsten Preferences-Utility zu.

3.1.2 Input

Wesentlich Interessanter wird es beim Input-Editor. Die Einstellungen der „Mouse-Speed", die „Double-Click" Zeit sowie die „Key Repeat" Einstellungen sind von 1.3 übernommen worden. Um sie kurz Zusammenzufassen: Mit „Mouse-Speed" läßt sich die Sensibilität der Mausbewegungen definieren: langsam (4) bis schnell (1). Die „Double-Click" Zeit ist der Zeitraum, innerhalb dessen ein „Double-Click" noch erkannt wird, „Key Repeat Delay" gibt an, nach welcher Zeit die Tastenwiederholung eingeschaltet wird und „Key Repeat Rate", wie schnell diese dann arbeitet. Neu aber ist die Möglichkeit, seine Einstellungen gleich auszuprobieren. Dazu gibt es das „Show"-Gadget, nach dessen Betätigung das rechts davon liegende Feld seine Farbe für genau den Zeitraum ändert, wie sie in „Double-Click" eingestellt haben, was weit anschaulicher ist als ein trockener Zahlenwert. Sie könne versuchen, dieses zu Doppelklicken; gelingt es Ihnen, wird daneben ein „Yes" erscheinen, wenn nicht ein „No". Auf diese Weise können Sie überprüfen, ob die gewählte Einstellung für Sie geeignet ist. Ebenso hat man in dem mit „Key Repeat Test" benannten Text-Gadget Gelegenheit, einen Text einzutippen und dabei die Tastaturwiederholung auszutesten. Doch auch der Input-Editor hat eine wesentliche Neuerung zu bieten: die Maus-

beschleunigung („Acceleration"). Einmal ist der verwendete Gadget-Typ an sich neu, denn wenn sie ihn anklicken, erscheint ein kleiner Haken darin, der anzeigt, daß die entsprechende Funktion angewählt wurde. Und auch die Funktion selbst ist bemerkenswert; die „Acceleration" bewirkt nämlich, daß der Mauszeiger sich bei kleinen Auslenkungen nur langsam bewegt, fährt man jedoch weitere Strecken, so wird er zunehmend schneller. Dies ist eine äußerst sinnvolle Neuerung, die bisher nur mittels zusätzlicher Hilfsprogrammen aus dem PD-Bereich installiert werde konnte und eine echte Bereicherung darstellt. Denn es ist damit möglich, sehr genaue Bewegungen auszuführen und dennoch schnell vom einem Ende des Bildschirms zum anderen zu kommen. Ganz gleich welche Mausgeschwindigkeit Sie bevorzugen ist dies also eine empfehlenswerte Funktion, die Sie mit Sicherheit bald nicht mehr missen möchten.

Im Input-Editor der Workbench 2.1 ist zusätzlich noch ein Auswahlfeld zur Festlegung der Tastaturbelegung vorhanden. Bei der WB 2.1 wird nämlich nicht mehr das gute alte „Setmap" irgendwo in der „Startup-Sequence" aufgerufen, sondern auch die Tastaturbelegung wie alle anderen Preferences behandelt. Die Bedienung dürfte keine Schwierigkeiten bereiten: Mit dem Rollbalken können Sie versteckte Tastaturbelegungen in den sichtbaren Bereich schieben und die gewünschte einfach anklicken, worauf sie in dem darunterliegenden Kästchen erscheint. Sonst bleibt alles beim alten.

3.1.3 Palette

Die Einstellung der Workbench-Farben ist nicht atemberaubend. Am oberen Ende können Sie eine der vier Farben wählen, die Sie zu verändern beabsichtigen, und dann

mit den drei Schiebereglern darunter Rot-, Grün- und Blauwert verändern. Viel läßt sich dazu nicht sagen, am Anfang muß man mit den Reglern wohl ein wenig spielen, bevor man das richtige Gefühl bekommt, um die Farbe, die man sich vorgestellt hat, auch auf den Bildschirm zu zaubern. Hilfreich erweist sich da das „Edit"-Menü, dessen Untermenü „Presets" einige Vorgaben enthält, die Sie einmal durchprobieren können. Wenn Ihnen dann eine Farbpalette besonders gut zusagt, können Sie diese ja Übernehmen oder gegebenenfalls noch leicht modifizieren.

3.1.4 WBPattern

Eine hübsche Neuerung von WB 2.0 sind die Hintergrundmuster (Pattern), die man mit diesem Voreinsteller wählen kann. Es ist möglich, unterschiedliche Muster vor das Workbench Fenster sowie die anderen Fenster zu wählen, wodurch das WB Fenster dann besser abgesetzt ist und man sich in dem Dschungel offener Windows nicht mehr so leicht verirrt. Je nachdem, für welchen Bereich Sie nun ein Muster wählen wollen, klicken Sie in einen der beiden Knöpfe links oben: den, hinter dem „Workbench" steht, der den mit „Windows" bezeichneten. Dies Knöpfe sind ebenfalls ein Novum und sind immer vertikal angeordnet. Sie schließen sich zumeist gegenseitig aus, so daß das Anwählen einer Option die anderen abwählt.

Doch zur Handhabung des Pattern-Editors selbst. Im mittleren Bereich sehen Sie eine Vergrößerung des aktuellen Musters, in der Sie auch mittels der Maus Veränderungen vornehmen können. Dazu wählen Sie eine Farbe aus der davon rechts liegenden Farbpalette, die dann an deren oberen Ende angezeigt wird. Anschließend klicken Sie einfach in das Editorfeld hinein, um einen Punkt in der gewählten Farbe zu setzen.

Halten Sie die Maus gedrückt, so können Sie freihändig zeichnen. Der jeweils letzte Linienzug wird dabei zwischengespeichert, so daß Sie, falls Ihnen ein Fehler unterlaufen ist, diesen Mit dem „Undo"-Gadget rückgängig machen können. Wie das Muster vor der letzten Veränderung aussah, wird zu diesem Zweck neben dem „Undo"-Feld angezeigt. Einen Eindruck, wie das Ganze dann am Ende aussehen wird, vermittelt Ihnen die Anzeige rechts oben: dort sind zwölf solcher Einheiten zusammengefügt, was auch hilft, die Ränder korrekt zu setzen, damit ein nahtloser Übergang garantiert ist. Für die weniger Kreativen sind wieder einige Vorgaben vorhanden, die Sie durch einfaches anklicken in den Editor übernehmen können.

Damit Sie nicht jedesmal den Editor mit „Use" verlassen müssen, um ein ganzes Window in seiner neuen Pracht zu sehen, ist es möglich, das gewählte Muster mittels „Test"-Gadgets in das (Workbench-) Fenster zu übernehmen. Wenn Ihr Entwurf vollends daneben ging, kann mit dem „Clear"-Gadget das Editorfeld schließlich noch komplett gelöscht werden, um nochmals von ganz vorne beginnen zu können.

3.1.5 Pointer

Auch bei Kick 1.3 war der Pointer-Editor bereits auf einen eigenen Bildschirm ausgelagert, der dem jetzigen auffallend gleicht. Für die Aufsteiger gibt es hier also nichts neues, für alle anderen ein paar kurze Erläuterungen:
Das Editieren, also Zeichnen an sich entspricht genau dem des Pattern-Editors. Darüber hinaus können Sie noch die Farbpalette für den Mauszeiger ändern, was genauso die das Ändern im Palette-Editor vonstatten geht. „Clear" löscht das Editorfeld und „Reset Color" stellt die gespeicherten Farben wieder her. Beachtung verdient nur

die „Set Point"-Funktion, mit der sie den Auslösepunkt des Mauszeigers festlegen können. Sollten Sie einmal keinen Pfeil als Mauszeiger wollen, sondern beispielsweise ein Fadenkreuz, so müssen Sie diesen Punkt genau in die Mitte des Kreuzes legen, denn schließlich wollen Sie ja damit auf die Objekte am Bildschirm zeigen. Dazu Klicken Sie auf „Set Point" und anschließend auf die entsprechende Stelle im Editorfeld. Daraufhin wird der gewählte Punkt umrahmt und Sie wissen, wo der neue Auslösepunkt sitzt.

3.1.6 Font

Mit dem Betriebssystem 2.0 ist es erstmals möglich, andere Zeichensätze als den Standard Topaz Font in der Workbench-Umgebung zu benützen. Für den Anwender von Disketten entsteht dabei aber das Problem, daß die entsprechenden Fonts nicht auf der Workbench Diskette, sondern auf der beiliegenden Fonts Diskette enthalten sind. Damit sind sie nicht von vornherein zugänglich, sondern müssen erst auf die Workbench Diskette kopiert werden. Dies wiederum aber geht nur vom Shell aus, außerdem fehlt es am nötigen Platz auf der Workbench. Für den Festplattenbenutzer bestehen diese Einschränkungen nicht, da ja die Daten aller Disketten bereits auf seine Platte kopiert wurden.

Es können verschiedene Fonts für Icon-Texte, Screen-Texte und System-Texte gewählt werden. Dazu wählen Sie den entsprechenden Knopf an und selektieren anschließend aus der Auswahlbox (gegebenenfalls auch durch verschieben des Rollbalken) den gewünschten Font, der dann in einem Test-Feld angezeigt wird. Bei der Workbench 2.1 ist die Vorgehensweise ein klein wenig anders. Dort existieren

keine Knöpfe, sondern Sie müssen nach dem Starten des Programms erst einmal den Verwendungszweck auswählen, bevor in einem weiteren Fenster die eigentliche Selektion stattfinden kann; zwei Fenster also statt einem. Außerdem wird im Font-Menü der WB 2.0 für jede Größe eines Fonts eine entsprechende Zeile ausgegeben, bei WB 2.1 gibt es zwei Kästen: einen für den Font und einen für die Größe. Lassen Sie sich aber dadurch nicht verwirren, sowohl die eine als auch die andere Anzeige ist einleuchtend, und da Sie die für Sie relevante Version ja sicher vor sich haben, werden Sie schon längst den Zweck der einzelnen Bildelemente verstanden haben.

Viel wichtiger ist, daß für die Icon-Texte zudem noch Vorder- und Hintergrundfarbe gewählt werden können. Dazu sind die „Text" und „Field" Farbpaletten zuständig. Bei Wahl einer Hintergrundfarbe mit dem „Text & Field" Knopf werden die Icon-Texte mit dieser Farbe unterlegt, wobei ein etwaiges Muster übermalt wird. Sollten Sie also eine Einbettung der Texte in den Hintergrund bevorzugen, ist es besser, „Text Only" zu wählen, als das Field-Feld auf die normale Hintergrundfarbe zu setzen, da auf diese Weise das Muster erhalten bleibt.

3.1.7 Screenmode

Da AmigaOS 2.0 mit dem Enhanced Chip Set (ECS) ja nun eine Vielzahl verschiedener Grafikmodi unterstützt und nicht nur HiRes / Lores und Interlace / Non Interlace wird ein eigener Vereinsteller nötig. Hier werden Ihnen die möglichen Bildschirmmodi für Ihre Hardware angezeigt und Sie können die bevorzugte wählen. Im wesentlichen hängt das Angebot davon ab, ob Sie ECS installiert haben oder nicht. Sie müssen außerdem unbedingt beachten, daß für einige Modi ein besonderer Monitor benötigt

wird, da der übliche 1081 / 1084 nicht flexibel genug ist um alle vom Amiga kommenden Signale zu verarbeiten. Im folgenden eine kurze Aufstellung der maximal zur Verfügung stehenden Modi:

Anzeigemodus	Bildschirmgröße (ohne Interlace)	Bildschirmgröße (mit Interlace)	Voraussetzungen
PAL: HighRes	640 x 256	640 x 512	PAL
PAL: SuperHires	1280 x 256	1280 x 512	PAL, ECS
NTSC: HighRes	640 x 200	640 x 400	NTSC, Fat Agnus

NTSC: SuperHires
1280 x 200
1280 x 400
NTSC, ECS

Productivity
640 x 480
640 x 960
Multiscan, ECS

A2024_10Hz
1008 x 800
n/a
A2024, ECS

A2024_15Hz
1008 x 800
n/a
A2024, ECS

Unter „Voraussetzungen" ist jeweils angegeben, ob Sie einen besonderen Monitor benötigen (Multiscan, A 2024) oder das ECS. Da der 1081 / 1084 sowohl PAL als auch NTSC darstellen kann, können Sie beide Auflösungen nützen. Sie müssen lediglich

den entsprechenden Monitor in den „Monitors" Ordner legen. Vorteil hoher Auflösungen ist natürlich, daß mehr Informationen dargestellt werden können. Allerdings unterstützen einige Programme die neuen Modi noch nicht. Und auch der NTSC Modus hat seinen Vorteil. Sie verlieren zwar 56 bzw. 112 Zeilen, doch dafür gewinnen Sie 10 Hz Bildwiederholfrequenz. Der PAL Modus läuft nämlich mit 50 Hz und der NTSC Modus mit 60 Hz. Und je höher die Bildwiederholfrequenz ist, desto ruhiger wird das Bild. Sollten Sie also vorhaben, längere Zeit vor dem Rechner zu sitzen und vielleicht Texte zu lesen oder zu schreiben, so tun Sie Ihren Augen mit dem NTSC Modus einen echten Gefallen. Allerdings funktioniert dieser erst mit dem FAT-Agnus Chip, und nicht mit dem ganz alten, nur 512 KByte Chip-Mem verarbeitenden Agnus 8371.

Sie erhalten zusätzliche Informationen über den jeweils gewählten Modus in dem mit „Properties of the Selected Mode:" betitelten Kasten. Dabei bedeutet „Draggable", daß Screens wie gewohnt herumgeschoben werden können, „Panelled", daß der Anzeigemodus aus mehren Teilflächen besteht (nur beim A 2024 interessant) und „Requires bypassing the Display Enhancer", daß ein evtl. vorhandener Flicker Fixer ausgeschaltet werden muß.

Eine weitere Verbesserung von AmigaOS 2.0 stellt die freie Definierbarkeit der Screen-Größe dar. Grundsätzlich werden als Default-Werte genau die gewählt, die auch die Auflösung angeben, also beispielsweise 640 x 256 in PAL:HighRes. Doch nun können diese Vorgaben verändert werden. Dadurch läßt sich der nutzbare Bereich verkleinern (nicht sinnvoll) oder auch vergrößern (schon viel mehr). Wenn Sie also in den entsprechenden Feldern 2000 x 1000 oder so eingeben, dann bekommen

Sie einen riesigen Bildschirm -natürlich wächst Ihr Monitor nicht, aber der Screen, auf dem ja die Windows liegen, wird größer. Da er nun nicht mehr auf den Monitor paßt, können Sie ihn ganz einfach verschieben, indem Sie mit der Maus in die Nähe des Randbereichs des Monitors fahren. Dazu müssen Sie allerdings die Funktion „AutoScroll" anwählen, sonst läßt sich der Screen nur mittels Menüleiste oder der „Mouse Screen Drag" Funktion verschieben (siehe: 3.1.12 IControl). Auch wenn nur die wenigsten von dieser Anwendung Gebrauch machen werden, sollten Sie es sich zumindest einmal ansehen: Es ist ein atemberaubendes Erlebnis (vor allem unter einem schnellen 68030). Zu guter Letzt kann auch die Anzahl der darstellbaren Farben verändert werden, da aber mehr als vier Farben kaum unterstützt werden und Sie zudem Geschwindigkeitseinbußen hinnehmen müßten, ist vom Gebrauch dieser Funktion abzuraten.

3.1.8 Overscan

Unter Overscan versteht man das Aufziehen des Bildschirms bis in die äußersten Ecken des Monitors. Daß dieser Bereich grundsätzlich nicht genützt wird, hat aber auch seinen guten Grund: Wenn Sie nicht gerade an einem Flatscreen Monitor sitzen, wird das Bild in den Ecken doch schon arg verzerrt. Wenn Sie aber auch das letzte Eck ausnützen möchten, so bietet sich dieser Voreinsteller an.
Dazu wählen Sie zunächst den Bildschirmmodus, für den Sie die Einstellungen vornehmen wollen (PAL, NTSC,...). Anschließend können Sie entweder den „Text Overscan" oder den „Standard Overscan" editieren. In beiden Fällen wird ein Bildschirm angezeigt, auf dem neun Quadrate verteilt sind. Mit den vier an den Seiten, können Sie, indem Sie sie mit der Maus ziehen, Höhe und Breite verändern. Mit den

vier in den Ecken lassen sich jeweils zwei aneinanderstoßende Ränder gleichzeitig verschieben. Und mit dem Quadrat in der Mitte kann der gesamte Screen verschoben werden.
Vor allem für Besitzer eines Multiscan Monitors ergibt sich eine interessante Variante: Wählen Sie aus den „Screenmodes" den NTSC Modus und vergrößern Sie mittels „Overscan" die Höhe des Bildschirms auf 232 Zeilen. Da beispielsweise der NEC 3D den Bildschirm in NTSC nicht auf volle Höhe auszieht, können Sie immer noch alle Bereiche des Screens sehen, also volle 232 Zeilen. Hierfür haben Sie nun 60 Hz Bildwiederholfrequenz. Ein guter Tausch für die 24 verlorenen Zeilen! Die 1081 / 1084 Besitzer können da leider nicht so viel machen, da dieser Monitor auch den NTSC Screen bildschirmfüllend darstellt und sich mit „Overscan" nicht mehr viel herausholen läßt.

3.1.9 Printer

Mit dem Printer-Editor kommen wir nun das erste Mal mit der Außenwelt in Kontakt. Hier läßt sich einer der Drucker auswählen, die Sie bei der Installation der Workbench 2.0 angegeben haben. Wie schon bei den Fonts gilt auch hier für die Diskettenbenutzer, daß im Normalfall erst einmal das Kopieren der Treiber von einer anderen Diskette auf die Workbench Diskette angesagt ist.
Wenn die Treiber aber erst einmal zur Verfügung stehen, dann haben Sie die freie Auswahl. Zusätzlich zur Einstellung des richtigen Druckers können anschließend auch noch Angaben zur Länge der verwendeten Papiers in Zeilen („Paper Length (Lines)", bei Endlospapier meist 72, bei DIN A4 66) und dem linken und dem rechten Rand („Left Margin (Chars)" und „Right Margin (Chars)") gemacht werden. Auch

Grundsätzliche Einstellungen, wie etwa den „Printer Port", der angibt ob Ihr Drucker an der parallelen (üblich) oder seriellen Schnittstelle angeschlossen ist, ob sie Endlospapier oder Einzelblatt verwenden („Paper type") und welche Breite das Blatt hat („Paper Size"). Von Anwendung zu Anwendung hingegen können die restlichen Parameter noch abweichen gemeint ist die zu verwendende Zeichendichte („Printer Pitch") die von normal („10-Pica") über schmal („12-Elite") bis hin zu sehr schmal („15-Fine") reichen kann. Für Briefe werden Sie sicherlich normal oder bestenfalls noch schmal verwenden Sollten Sie jedoch Tabellen mit vielen Spalten ausdrucken wollen, so können sich die 136 Zeichen pro Zeile, die eine sehr schmale Einstellung auf DIN A4 bietet, durchaus bewähren. Mit „Print Spacing" ist schließlich noch der Druckabstand, also die Anzahl der Zeilen die auf einen Zoll (2,54 cm) passen gemeint. Wann Sie bei der „Print Quality" die Schönschrift („Letter") oder Schnellschrift („Draft") bevorzugen dürfte auf der Hand liegen.

3.1.10 PrinterGfx

Da sicher eines Tages der Wunsch nach etwas mehr als nur ödem Text auf dem Papier auftaucht, können Sie mit dem PrinterGfx-Editor die nötige Konfiguration zum Grafikausdruck einstellen. Da wäre zunächst einmal die Positionierung der Grafik auf dem Papier. Da, abhängig vor der Größe des auszudruckenden Bildes, meistens nicht die volle Breite auf dem Blatt ausgenutzt wird, können Sie unter „Left Offset" entweder den absoluten Abstand zum linken Rand in 1/10 Zoll, (also 2,54 mm) angeben, oder „Center Picture" anwählen, wodurch der Ausdruck zentriert wird. Über die Abmessungen des Ausdrucks können Sie in den Feldern „Width" und „Height" Angaben machen. Diese werden nach dem Zustand des „Type" Blättersymbols interpretiert. „Ignore"

ignoriert die Angaben, „Bounded" legt maximale Grenzwerte fest, „Absolute" erzwingt eine Anpassung an die angegebenen Maße, ebenso „Pixels", nur daß in diesem Fall die Werte in Punkten angegeben sind, und nicht in Zoll. „Multiply" schließlich sorgt für eine Vergrößerung der Grafik um den angegebenen Faktor. Bei den letzten drei Typen kann der Ausdruck durch explizite Angabe von Höhe und Breite stark verzerrt werden. Deshalb ist es in diesen Fällen eher ratsam, einen der beiden Werte auf Null zu setzen, wodurch eine automatische Anpassung an den anderen Wert vorgenommen wird. Um z.B. ein 6 Zoll breites Bild zu erzwingen, das entsprechend seiner Abmessungen ausgedruckt und nicht verzerrt werden soll, so wählen Sie einfach „Absolute", setzen „Width" auf „6" und „Height" auf „0" und das wars schon! In diesem Zusammenhang kann man auch gleich das „Scaling"-Gadget betrachten, das die Optionen „Fraction" und „Integer" zur Verfügung stellt. Wählt man erstere, so kann stufenlose vergrößert werden. Bei letzterer geht dies nur in Vielfachen der Größe der Grafik, die man ausdrucken will. Diese Einstellung ist immer dann sinnvoll, wenn Bilder mit Linien oder ähnlichem ausgedruckt werden sollen, da so beispielsweise unterschiedliche Strichdicken vermieden werden.

Doch das waren noch lange nicht alle zur Verfügung stehenden Optionen. Sollten Sie über einen Farbdrucker verfügen, so könnte es sein, daß die Farben auf dem Ausdruck nicht so ganz mit denen auf dem Bildschirm übereinstimmen. Um dieses Problem zu korrigieren, gibt es die „Color Correct"-Gadgets. Sie können mit ihnen die Anzahl der Farben aus dem Rot-, Grün- und Blaubereich verringern, was den Ausdruck durchaus verbessern kann. Welche Einstellungen im jeweiligen Fall am besten sind, hängt jedoch sehr stark vom verwendeten Drucker ab und Sie werden wohl einige Probe-

ausdrucke investieren müssen, bis Sie die richtigen Einstellungen gefunden haben. Wenn nur ein schwarz-weiß Drucker zur Verfügung steht, so müssen die Farben irgendwie umgerechnet werden. Die Einstellung „Color" im „Shade"-Feld scheidet schon einmal aus. Doch da gibt es ja noch „Grey Scale 1" und „Grey Scale 2" sowie „Black & White". Die letzte Option bewirkt eine Aufteilung aller Farben auf entweder schwarz oder weiß. Wo die Grenzlinie zu ziehen ist, läßt sich mit dem „Treshold"-Regler steuern. Dies ist beispielsweise dann notwendig, wenn Sie ein sehr helles oder aber sehr dunkles Bild ausdrucken wollen, da sonst womöglich ein einfarbig weißes bzw. schwarzes Blatt aus Ihrem Drucker wandert. Noch besser sind da schon die Graustufenumrechnungen. Da ein Drucker meist eine höhere Auflösung bietet als ein Monitor, kommen auf einen Bildpunkt auf dem Bildschirm mehrere auf dem Ausdruck. Je nachdem wie viele dieser Punkte gesetzt und in welchem Muster sie angeordnet sind, können später unterschiedliche Graustufen ausgemacht werden. „Grey Scale 2" ist dabei eher zu vernachlässigen, da es nur vier Farben verarbeiten kann und für Benutzer eines A 2024 Monitors gedacht ist. Wählen Sie daher am besten „Grey Scale 1".

Sowohl für Farbdrucker als auch für solche, die nur Schwarzweiß zu Papier bringen ist das „Dithering" interessant. Da auch Farbdruckern nur eine begrenzte Anzahl von Farben zur Verfügung steht (meist drei oder vier) müssen die restlichen durch Mischen erzeugt werden - und nichts anderes passiert auch bei der Graustufenumrechnung, nur hier werden lediglich die Farben schwarz und weiß gemischt. Zur Verfügung stehen die Algorithmen „Ordered", „Halftone" und „Floyd-Steinberg". Gerade die letzte verschluckt eine Menge Rechenzeit, liefert aber meistens auch die anschaulichsten

Ergebnisse. Auch in diesem Fall ist wieder einmal Ausprobieren angesagt.
Was noch übrig bleibt, sind einige ebenso wichtige wie leicht verständliche Selektionsmöglichkeiten. So wählen Sie mit „Aspect" ob die Grafik um 90∞ gedreht werden soll, also praktisch vertikal ausgegeben wird, oder aber wie üblich horizontal. Mit „Image" = „Negativ" vertauschen Sie schwarz mit weiß, was gerade bei dunkler Grafik enorm das Farbband schont. Bei „Positive" erhalten Sie den Ausdruck so, wie das Bild auf dem Monitor erscheint. Einen anderen Weg, Ihr Farbband zu schonen, oder aber die Qualität der Ausdrucke zu erhöhen, stellt der „Density"-Regler dar. Er gibt an, wie „dicht" die Pixel beim Ausdruck angeordnet werden, was zu einer gewaltigen Verbesserung führen kann, jedoch auch den Druck verlangsamt. Auch muß man berücksichtigen, daß nicht jeder Drucker die hohen Einstellungen verarbeiten kann. Als letzte Möglichkeit den Ausdruck aufzupeppen, bietet sich die „Smoothing"-Option an. Die bewirkt, daß schräge Linien geglättet werden, wodurch die Auflösung scheinbar ansteigt. Eine kleine Einschränkung existiert jedoch: Sie können „Smoothing" nicht mit „Floyd-Steinberg" kombinieren.

Allerdings bleiben immer noch genügend Kombinationsmöglichkeiten übrig, um Ihnen so viele Probedrucke abzuverlangen, daß von einem Blitzeinstieg keine Rede mehr sein kann. Deshalb an dieser Stelle eine Variante, die sich bewährt hat und die Sie ja immer noch Ihren Wünschen (und Ihrem Drucker) anpassen können:

Dithering
Halftone
Shade
Grey Scale 1

Scaling
Integer
Smoothing
off

Image
Positive
Center Picture
on

Aspect
Horizontal
Limits
Ignore

Außerdem sollten Sie noch „Density", je nachdem wie lange Sie warten wollen und was Ihr Drucker so verkraftet, größer als eins wählen.

3.1.11 Serial

Mit dem Serial-Editor kommen wir wieder zu einer einfacheren Angelegenheit. Es geht dabei um die Konfiguration des seriellen Ports, was in erster Linie wichtig ist, wenn man ein Modem daran angeschlossen hat. Aber auch in diesem Fall werden die Einstel-

lungen oftmals vom verwendeten Terminalprogramm vorgenommen, so daß die Grundeinstellungen der Workbench zumeist völlig belanglos sind. Sollte man jedoch den Amiga beispielsweise über Nullmodem mit einen zweiten Rechner verbunden haben, auf dem man dann mittels AUX-Handler ein Remote-Shell eröffnen will, so wird auf eben diese Parameter zurückgegriffen.

Die wichtigste Einstellung ist sicher die Übertragungsrate, oder „BAUD Rate". Sie läßt sich in den üblichen Schritten konfigurieren, wobei aber völlig unverständlich ist, warum der letzte Schritt, die 38400 Baud nicht unterstützt werden. Der Amiga ist mittels des „serial.device" nämlich durchaus zu solchen Raten in der Lage! Die unüblichen 31250 Baud werden ausschließlich für MIDI und nicht zur Rechnerkopplung genützt. So bleiben für eine Remote-Shell meist nur 19200 Baud übrig, was für diesen Zweck aber auch ausreicht. Sie müssen unbedingt beachten, daß diese und auch alle anderen Einstellungen auf jeden Fall mit denen des angeschlossenen Zweitrechners bzw. der Mailbox übereinstimmen! Lediglich die „Input Buffer Size" können Sie frei wählen. Auch hier gilt: je mehr, desto besser. „Parity", „Bits / Char" und „Stop Bits" sind weitere entscheidende Parameter, die üblicherweise in drei Zeichen abgekürzt werden. So bedeutet 8N1 (was heutzutage am meisten Verbreitung gefunden hat) 8 „Bits / Char", „None" „Parity" und 1 „Stop Bits".

Das „Handshaking" hängt vom verwendeten Modem bzw. Kabel ab. So verwenden High-Speed Modems grundsätzlich „CTS/RTS", wärend ältere Typen noch auf das problematischere „XON/XOFF" zurückgreifen müssen. Bei Verwendung eines Nullmodem Kabels ist im Grunde überhaupt kein „Handshaking" nötig (also „None");

sollten Sie jedoch eines der besseren Kabel haben, d.h. ein Kabel, bei dem auch die CTS- und RTS-Leitungen belegt sind, so sollten sie die entsprechende Option wählen.

3.1.12 IControl

All die Einstellungen, die in kein anderes Preferences-Programm gepaßt haben, finden Sie in „IControl". Dementsprechend weit gestreut sind auch die Themenbereiche. Gleich die erste Knopfleiste ist die sinnloseste überhaupt, weshalb sie bei der Workbench 2.1 auch wieder entfernt wurde. Unter dem Titel „Verify Timeout" soll dem Anwender Gelegenheit gegeben werden, festzulegen, nach welchem Zeitraum das System mit seiner Bearbeitung fortfährt, wenn es auf die Reaktion eines Programms vergeblich wartet. Im Klartext bedeutet das: Sollte ein Programm derart fehlerhaft programmiert sein, daß es das System warten läßt „bis es schwarz wird" so können Sie einen Stillstand dadurch vermeiden, daß Sie das System zwingen, nach einer gewissen Zeit dennoch weiterzumachen. Daß das natürlich nicht die Lösung aller Probleme sein kann, dürfte klar sein, denn früher oder später wird diese rüde Haltung des Systems Spuren hinterlassen, und zwar bevorzugt solche in roten Kästen am Kopf des Bildschirms. Wie dem auch sei: Vergessen Sie diese Einstellung!

Sinnvoller werden da schon die Definition von Tastaturkürzeln, die bisher nicht veränderbar waren. So können Sie unter „Command Keys" die Tastenkombinationen zum Verlegen des Workbenchscreens und Beantworten von Requestern nach eigenen Vorstellungen gestalten. Da allerdings recht viele Anwendungsprogramme ihre eigenen Kürzel für verschiedene Sonderfunktionen haben, und nur die bisher nicht veränderbaren Kominationen unbelegt ließen, ist das vielleicht gar nicht so sinnvoll! Denn ob ich nun

Amiga-N drücke, oder Amiga-S ist doch auch schon egal, oder? Wohl aus diesem Grund kann auch diese Einstellung unter Kick 2.1 nicht mehr vorgenommen werden, weshalb Sie sich jetzt nicht mehr an eigene Abkürzungen gewöhnen sollten.

Anders ist es mit dem „Mouse Drag". Diese Funktion ist endlich wieder einmal sinnvoll. Denn wenn man bisher mit dem Mauszeiger immer auf die Menüzeile eines Screens fahren mußte, um diesen zu verschieben, so geht das jetzt auch einfacher: Sie betätigen einfach die unter „Mouse Drag" festgelegte Taste und dann den linken Mausknopf - und schon können Sie den gesamten Bildschirm verschieben, egal wo sich ihr Mauszeiger befindet. Diese Funktion ist auch in Kombination mit einem übergroßen Bildschirm (siehe auch: 3.1.7 ScreenMode) äußerst hilfreich, falls sie nicht gerade „AutoScroll" eingeschaltet haben. Denn nur so können Sie in diesem Fall in die entlegenen Bereiche des Bildschirms fahren, da die Menüleiste möglicherweise gar nicht sichtbar ist. Wegen des selteneren Gebrauchs der Ctrl-Taste ist diese hierfür gut geeignet, auf jeden Fall vermieden werden sollte es, nur die Shift-Taste alleine einzusetzen, da sonst das Auswählen mehrere Icons logischerweise nicht mehr funktioniert.

Ebenfalls für den Anwender übergroßer Bildschirme ist die Funktion „Screen menu snap" gedacht. Da die Menüs grundsätzlich von links her in die Menüleiste gefüllt werden, kann es bei entsprechend großem Bildschirm schon einmal vorkommen, daß sie im momentanen Ausschnitt nicht erreichbar sind. Wurde jedoch „Screen menu snap" gewählt, so schaltet die Anzeige, solange Sie den rechten Mausknopf gedrückt halten, ins linke obere Eck, so daß Sie ohne von Hand nach oben fahren zu müssen, den gewünschten Menüpunkt auswählen können. Lassen Sie den rechten Mausknopf

wieder los, springt die Anzeige wieder auf den vorher betrachteten Ausschnitt zurück. Einfacher geht's kaum.

Auch die Option „Text gadget filter" hat ihre Berechtigung. Es gibt ja einige Tastenkombinationen, die in Textfeldern Funktionen bewirken anstatt in den Text übernommen zu werden. Z.B. Ctrl-X zum löschen der Zeile, die gerade editiert wird. Diese Zeichen werden sinnvollerweise ausgefiltert, denn was würde es nützen, eine Zeile mit Ctrl-X zu löschen um dann anschließend ein Ctrl-X Steuerzeichen im Textpuffer vorzufinden. Sollte jedoch einmal die Notwendigkeit bestehen, dieses oder andere verwendete Steuerzeichen in ein Textfeld zu übernehmen, so können Sie die Filterfunktion ausschalten, woraufhin selbstverständlich auch die Sonderfunktionen nicht mehr funktionieren.

Die letzten Einstellungen, überschrieben mit „Coercion", beziehen sich ausschließlich auf den Productivity-Modus. Da der Monitor diesen in einer anderen Frequenz darstellt, kann es zu Problemen kommen, wenn neben dem Productivity-Screen auch noch ein Screen in einer der normalen Auflösungen dargestellt werden soll. Während der Anzeige kann nämlich auch ein Multiscan Monitor die Frequenz nicht ändern, was zur Folge hat, daß ein Teil des Bildschirms verzerrt würde. Deshalb versucht der Amiga, mittels des Interlace-Modus und unter Verlust von Farben das Bild so gut es geht noch darzustellen. Das aber wiederum bedeutet, daß der Bildschirm plötzlich flickert und die Farben durcheinanderkommen. Diese Effekte können Sie mit „Avoid flicker" und „Preserve colors" verhindern, nehmen dafür aber möglicherweise ein umso stärker verzerrtes Bild in Kauf.

3.1.13 Sound (WB 2.1)

Die nun folgenden Voreinsteller sind erst ab der Workbench 2.1 enthalten. Und auch wenn sie sich bis zu deren endgültigen Erscheinen noch ändern können, sollen sie an dieser Stelle besprochen werden. Da ab WB 2.1 auch die deutsche Sprache unterstützt wird, werden für den Rest der Besprechung der Voreinsteller die deutschen Begriffe für Texte, Gadgets oder ähnliches verwendet.

Sie kennen ja das beliebte Bildschirmblitzen, das in Ermangelung eines internen Piepsers wie er von PCs her bekannt ist, den Anwender auf irgend etwas aufmerksam zu machen versucht. Da aber allein wegen der hervorragenden Soundeigenschaften fast jeder seinen Amiga an irgendeine Art Verstärker angeschlossen hat, kommt nun doch auch das Verlangen nach einer entsprechenden Funktion auf. Da wir aber von einem Amiga reden, begnügen wir uns nicht mit einem lapidaren „Pieps", sondern da muß schon mehr her. Doch zunächst einmal, können Sie „Anzeige blitzen" und „Ton ausgeben" getrennt voneinander an- oder ausschalten. Haben Sie sich für einen Ton entschieden, steht die Wahl zwischen einem einfachen „Piepsen" oder aber „Digitalis. Sound" frei.

In beiden Fällen können Sie „Lautstärke" und „Tonhöhe" mittels Schieberegler einstellen, im Falle des Piepsens auch noch die „Piepslänge". Sollten Sie aber ein gutes Sample bevorzugen (möglicherweise etwas in der Art: „He, aufwachen!"), dann erhalten Sie über „Sound auswählen..." einen Filerequester, in dem Sie nach dem entsprechenden Filenamen suchen können. Zu guter Letzt läßt sich der Sound auch

noch testen, damit Sie nicht erst in die Verlegenheit kommen den Rechner so lange quälen zu müssen, bis er freiwillig ein Bildschirmblitzen von sich gibt.

3.1.14 PrinterPS (WB 2.1)

Der Voreinsteller zu Postscript-Druckern ist im Prinzip eine Zusammenfassung von „Printer" und „PrinterGfx", bezogen auf Postscript. Die wesentlichen Elemente finden Sie auch in den genannten Abschnitten in diesem Buch erklärt. Vielleicht erwähnenswert ist die Einstellbarkeit der „Kopien", da bei Postscript-Druckern das Verfahren des wiederholten Ausdrucks ja ganz anders ist als bei anderen Druckern. Hier werden nicht nochmals alle Daten zum Drucker geschickt, (was auch ziemlich aufwendig wäre, da bei Postscript ja erst einmal die komplette Seite aufgebaut werden muß), sondern die sich noch im Speicher des Druckers befindliche Seite einfach nochmals ausgegeben. Außerdem können Sie mittels der DPI-Einstellungen noch die Auflösung des Drucks bestimmen, was in etwa dem „Density"-Regler der normalen Druckertreiber entspricht.

3.1.15 Locale (WB 2.1)

Das Herzstück der Workbench 2.1 ist die sogenannte „localization", also die Konfiguration des Systems an das gewünschte Land. Dies beinhaltet in erster Linie die Sprache, aber auch solche Kleinigkeiten wie Landeswährung oder das Format von Zeit und Datum. Im Moment berücksichtigt diese Einstellungen nur das Betriebssystem selbst. Doch wenn in Zukunft neue Programme auf den Markt kommen, wird es kein Nachblättern im Wörterbuch mehr geben, um den Sinn der Funktionen zu verstehen,

sondern Sie werden erst gar nicht merken, daß es sich um kein deutsches Produkt handelt.
Doch was müssen sie denn nun tun, um aus Ihrem Amiga deutsch beizubringen. Es ist geradezu lächerlich einfach. Da Sie bei der Installation wahrscheinlich schon die deutsche Variante - und nur diese - ausgewählt haben, sind in dem Feld „Verfügbare Sprachen" auch nur „Deutsch" und „English", das immer zur Verfügung steht, aufgeführt. Klicken Sie einfach auf „Deutsch", worauf dieses in dem Kasten „Bevorzugte Sprachen" erscheint. Bevorzugt deshalb, weil es ja immer sein kann, daß ein Programm gerade die von Ihnen gewählte Sprache nicht unterstützt, woraufhin die nächste Sprache aus der Liste gesetzt wird. Wird überhaupt keine der bevorzugten Sprachen unterstützt, wird in der Standardsprache des Programms fortgefahren. Deshalb ist es auch gar nicht so sinnlos, nach „Deutsch" auch noch „English" anzuwählen, denn es kann ja sein, daß ein Programm nur seine Landessprache und eben Englisch unterstützt. Allerdings sollten die zukünftigen Programme alle möglichen Sprachen beherrschen, so daß dieser Fall eigentlich nie auftreten sollte. Um die Liste der bevorzugten Sprachen zu löschen, kann man das mit „Sprachen löschen" bezeichnete Gadget betätigen.
Zur weiteren Wahl landesspezifischer Einstellungen, können Sie unter „Land" noch Ihr Heimatland anklicken, was Ihnen in Zukunft so ungewöhnliche Datumsformate wie 02-17-92 erspart und dafür 17.02.92 angezeigt wird. Und schließlich kann noch die zutreffende Zeitzone gesetzt werden, indem Sie auf Ihre momentane Position in der Weltkarte klicken. Da diese ein wenig klein geraten ist, sollte vielleicht darauf hingewiesen werden, daß beispielsweise Deutschland bei +1 Stunde gegenüber GMT liegt. Dummerweise wird das „+" nicht mit angezeigt, aber so weit daneben werden Sie schon nicht klicken.

3.2 Devices (WB 2.1)

Neben den Preferences sind die Device ganz entscheidend mitverantwortlich für die Konfiguration Ihres Rechners. Doch warum steht dann dieses störende (WB 2.1) in der Überschrift? Schließlich haben die Devices schon in den ersten Kickstart-Versionen existiert. Nun, das liegt daran, daß wir in diesem Abschnitt des Blitzeinstiegs uns noch immer auf Workbench-Ebene befinden, und die Devices erst ab der WB 2.1 in die Workbench intergriert wurden. So ist es sicher nicht verkehrt, auch als WB 2.0 Besitzer diesen Abschnitt durchzulesen, doch die besprochenen Ordner und Icons existieren auf der WB 2.0 eben noch nicht. In der Funktionsweise selbst hat sich nichts geändert, lediglich in der Erscheinung und auch ein klein wenig in der Sruktur.

3.2.1. DOS Drivers

Nach dem Öffnen des „Devs" Ordners sehen Sie vier weitere Schublden vor sich. Der erste ist mit „DOSDrivers" überschrieben und enthält prinzipiell das, was bis einschließlich WB 2.0 in der „Mountlist" steht. Da stellt sich natürlich die Frage , warum man diese Treiber, dem Anwender denn so lange vorenthalten hat, und wozu sie überhaupt da sind. Nun, prinzipiell benötigt man Treiber um auf gewisse Geräte (nicht anderes heißt „device") zugreifen zu können. So gibt es beispielsweise das trackdisk.device, das für die Datenübertragung von und zu den Diskettenlaufwerken zuständig ist. Da jeder Amiga mindestens ein Diskettenlaufwerk besitzt, ist dieses „Gerät" kein Thema, es muß immer vorhanden sein. Schon anders sieht es mit der RAD, der resetfesten RAM-Disk aus. Nicht jeder hat genügend Speicherplatz, um sie betreiben zu können, oder auch gar keinen Bedarf. Andere wiederum benötigen sie ständig. Und genau da setzt nun die WB 2.1 ein: Wenn Sie die RAD benötigen, so brauchen Sie nichts weiter zu tun, als das mit RAD betitelte Icon in den „DOSDrivers"-Ordner zu legen und schon ist sie da!

Wollen Sie keine RAD, so schmeissen Sie das Ding einfach wieder raus, einfacher geht's kaum! Doch wohin mit dem Ding, bzw. noch wichtiger: woher? Genau um diese Frage zu klären, sollten Sie einmal den Ordner namens „Storage", der sich ebenfalls im Hauptverzeichnis befindet öffnen. Und, oh Wunder, in ihm sind alle vier Schübe des „Devices" Ordners noch einmal vorhanden. Die stehen da natürlich nicht nur herum, um Ihre Festplatte / Diskette noch voller zu machen, sondern, wie der Name „Storage" schon ankündigt, sind sie als Aufbewahrungsort für nicht benötigte Devices gedacht. Und aus dem zweiten „DOSDrivers"-Schub können Sie eben die Treiber holen, die Sie benötigen, bzw. die, die Sie nicht benötigen ablegen, für den Fall daß Sie es sich später

einmal anders überlegen. Offenbar traut man erst jetzt dem Anwender zu, derartige Entscheidungen zu treffen, denn die nötigen Anweisungen im Shell, um den gleichen Effekt bei älteren Workbench-Versionen vorzunehmen, sind schon ein Stück umständlicher.

Doch jetzt wollen wir erst einmal sehen, was uns alles zur Auswahl steht. Da ist also zunächst einmal die schon angesprochene RAD, die etwa 900 KByte Speicher schluckt, dafür aber ein den Diskettenlaufwerken voll kompatibles Speichermedium im RAM erzeugt. Sehr sinnvoll ist Sie beispielsweise beim Installieren von Software von Diskette auf Festplatte. Da kann zuerst mittels Diskcopy die komplette Disk in die RAD kopiert werden, um die Installation dann aus dem RAM durchzuführen. Dies bringt oftmals einen enormen Geschwindigkeitsgewinn, da das Diskcopy recht schnell Spur für Spur einließt, während bei der Installation selbst meist ständig die gleichen Spuren wieder und wieder gelesen werden müssen.

Dann haben wir den mit „speak" überschriebenen Treiber zur Sprachausgabe, der eigentlich nicht schaden kann, wenn er im „Devs"-Ordner landet, da er eigentlich kaum Speicher frißt und, solange er nicht angesprochen wird, auch sonst nicht weiter stört. Stattdessen gehen Sie sicher, daß Programme, die unbedingt diese grausame Art der Kommunikation anwenden wollen, dann auch funktionieren.

Ebenso sollte man nicht auf den „pipe"-Treiber verzichten, der vor allem im Shell eine große Bedeutung hat (und wieder einmal: siehe Shell). Da auch von der Workbench gestartete Programme möglicherweise Script-Dateien starten, die auf die „Pipe" zugreifen können, ist es ratsam ihn immer bereit zu haben.

Weniger interessant ist da schon der „AUX"-Treiber, denn der ist nun wirklich für Sonderanwendungen vorbehalten. Er behandelt die serielle Schnittstelle, aber auch nicht in allen Bereichen, sondern nur dann, wenn über den seriellen Port ein Shell-Fenster geöffnet wird. Wenn Sie also nicht gerade ein fortgeschrittener Anwender sind (und wenn doch, warum lesen Sie dann dieses Buch?), werden Sie ihn wohl nicht benötigen.

Doch nun, bis zum Schluß aufgespart, kommt der zweite Hammer, den unser neues Betriebssystem zu bieten hat. Denn hinter den unscheinbaren Namen „PC0" und „PC1" befinden sich die Treiber zum Lesen und Beschreiben von Disketten im MS-DOS Format. Wenn Sie also gelegentlich mit PCs beschriebene Disketten weiterverarbeiten wollen, um z.B. Textfiles zwischen den unterschiedlichen Rechnertypen auszutauschen, so ist nichts weiter zu tun, als die Treiber in den „Devs"-Ordner zu legen. Anschließend können Sie eine MS-DOS formatierte Diskette in ein ganz normales Amiga-Drive einlegen und sie über PC0: bzw. PC1: ansprechen. Dabei ist mit PC0: das Laufwerk, das normalerweise mit DF0: angesprochen wird, gemeint, und entsprechend PC1: das DF1:-Drive. Sie können auch unformatierte Disketten auf MS-DOS Format formatieren, und zwar über den ganz normalen „Format"-Befehl. Wenn Sie PC0: oder PC1: angegeben haben, wird in dem entsprechenden Laufwerk eine MS-DOS Diskette erzeugt. Allerdings muß auch erwähnt werden, daß Sie selbstverständlich keine HD-Disketten bearbeiten können, sondern nur das dem Amiga 880 KByte entsprechendem 720 KByte Format verwenden. Doch die Hauptsache ist ja, daß überhaupt ein Datenaustausch stattfinden kann, und noch dazu so einfach.

Das waren sie also, die zur Verfügung stehenden Treiber. Wenn für andere Anwendungen einmal Treiber mitgeliefert werden, muß entsprechend vorgegangen werden, d.h. die Treiber einfach in den „DOSDriver"-Schub des „Devs"-Ordners legen. Bitte bedenken Sie auch, daß irgendwelche Veränderungen der Inhalte dieses Schubs erst NACH einem Systemneustart (Reset) aktiv werden. Wenn Sie also mal schnell eine resetfeste RAM-Disk benötigen, ist es der falsche Weg, das Icon kurz in den richtigen Ordner zu ziehen. Vielmehr müssen Sie das entsprechende Icon Doppelklicken, um den Treiber vorübergehend (d.h. bis zum nächsten Systemneustart) zu aktivieren. Dadurch ist es natürlich auch viel einfacher, einen nur sehr seltenen Treiber kurzzeitig einzubinden, ohne ihn erst durch die Gegend zu schieben.

3.2.2 Keymaps

Bei den „Keymaps" läuft die Sache ein wenig anders. Hier bedeutet die Existenz von Tastaturbelegungen im „Devs"-Ordner nicht, daß diese alle aktiviert werden - es kann ja immer nur eine Einstellung gültig sein. Stattdessen werden alle hier befindlichen Tastaturbelegungen in dem Preferences-Programm „Input" zur Auswahl gestellt. Der „Keymaps"-Ordner im „Storage"-Verzeichnis hat deshalb eigentlich keine Bedeutung, denn die letzte Entscheidung, welche Tastaturbelegung Sie nun aktivieren, läuft über die Preferences.

3.2.3 Monitors

Bei den „Monitors" ist es ebenso. Sie wählen nur die möglichen Monitoreinstellungen, abhängig von Ihrem Monitortyp. Das hat noch überhaupt keine Auswirkungen. Erst im

„Screenmode"-Voreinsteller werden abhängig davon mehr oder weniger Bildschirmmodi angezeigt. So sollten Sie als Besitzer eine 1081 / 1084 sowohl den „PAL"-, als auch den „NTSC"-Monitor in den „Monitors"-Ordner des „Devs"-Verzeichnisses ziehen, da dieser Monitortyp beide Einstellungen verkraftet. Legen Sie allerdings auch noch den „Multiscan" dazu, und besitzen darüber hinaus noch das ECS, so sollten Sie sich davor hüten, die neue Auswahlmöglichkeit bei den „Screenmodes" auszunützen und den „Productivity"-Modus einstellen. Denn das Programm geht ja dann davon aus, daß Sie über einen Multiscan Monitor verfügen. Nicht benötigte Monitortypen können wieder im „Storage" gelagert werden.

Besitzer der Workbench 2.0 haben auch schon Gelegenheit, „Screenmode" mit den nötigen Informationen zu versorgen, und das geht praktisch genauso, wie eben beschrieben. Lediglich die Namen der Verzeichnisse unterscheiden sich. Hier sind die entsprechenden Ordner direkt im Hauptverzeichnis und nennen sich „Monitors" für die verfügbaren Typen und „MonitorStore" für alle nicht vorhandenen.

3.2.4 Printers

Das bei den „Keymaps" gesagte entspricht genau der „Printers"-Schublade. Auch hier treffen Sie die Auswahl welcher Druckertreiber letztendlich aktiviert wird über ein Preferences Programm, diesmal im „Printer"-Editor. Da bei der Installation aber nur recht allgemeine Druckertypen ausgewählt werden können, bekommen Sie meist mehr Druckertreiber installiert, als Sie letztendlich wollten. So werden bei der Wahl von

HP-Druckern immerhin fünf verschiedene Treiber kopiert, vom HP Deskjet bis zum HP Laserjet. Die Typen die Sie nicht benötigen, können wieder in den „Storage"-Ordner wandern - oder auch gelöscht werden (natürlich nur auf der Festplatte, bzw. Sicherheitskopie).

4. Utilities und Tools

4.1 Workbench Utilities

4.1.1 Clock

Da nun die primären Installations- und Konfigurationsarbeiten abgeschlossen sind, ist es an der Zeit auch einmal etwas sinnvolles mit dem neuen Betriebssystem anzufangen. Und was gibt es da für uns Digitaluhrenträger besser geeigneteres als die „Clock". Wenn Sie dieses Programm also eine Zeit lang genossen haben, und fasziniert verfolgt haben, wie Ihre Lebensuhr immer weiterläuft, sollten wir vielleicht fortfahren,

denn sonst war das wohl nichts mit dem „Blitzeinstieg". Es gibt auch nichts neues zu sehen, da das „Clock"-Utility bereits unter WB 1.2 existierte. Für die Frischlinge sei nur noch schnell auch die Menüleiste hingewiesen, die noch einige mehr oder weniger sinnvolle Optionen zu Verfügung stellt.

4.1.2 More

Mit „More" läßt sich schon eher etwas anfangen, denn das Lesen von Texten und vor allem Anleitungen ist eine der wichtigsten Arbeiten des gewissenhaften

Computeranwenders. Es bestehen zwei Möglichkeiten, dieses Programm zu starten. Entweder, Sie klicken zuerst einmal auf das File, das Sie lesen möchten (es sollte sich dabei selbstverständlich um ein Textfile handeln), halten dann die Shift-Taste gedrückt und Doppelklicken dabei das „More"-Icon. Dadurch wird der gewählte Text gleich mitgeladen. Startet man „More" jedoch, ohne vorher ein anderes Icon angewählt zu haben, so öffnet sich nicht nur das Textfenster sondern auch ein Filerequester, mit dessen Hilfe man dann den Text auswählen kann, den man gerne gelesen hätte.
Bei dieser Gelegenheit bietet es sich an, auch gleich den Standard-Filerequester zu erklären. Denn nachdem vor AmigaOS 2.0 jeder Programmierer seinen eigenen Filerequester programmieren mußte und die Ergebnisse auch dementsprechend differierten, gibt es jetzt einen Standard, der auch für die Entwickler neuer Software leicht anzuwenden ist, weshalb davon ausgegangen werden kann, daß Sie sich nie wieder an einen anderen Filerequester gewöhnen müssen.

Da wäre zunächst einmal ein Feld, in dem das gewählte Inhaltsverzeichnis erscheint. Den angezeigten Ausschnitt können Sie selbstverständlich mittels des Rollbalkens und der Pfeile am rechten Rand verschieben. Darunter gibt es ein Feld namens „Pattern", in dem ein Muster für die anzuzeigenden Filenamen angegeben ist. Dies entspricht den AmigaDOS-Konventionen und sollte bereits von älteren Betriebssystem-Versionen bekannt sein. Die wichtigsten beiden Muster sind in diesem Zusammenhang sicherlich „#?" und „~". Ersteres ist ein Muster, das auf alles zutrifft und letzteres invertiert die gewählten Files. So können mit „~(#?.info)" alle Files mit Ausnahme derer, die mit „.info" enden, angezeigt werden. Wieder eine Zeile tiefer folgt nun der Pfadname und drunter dann der eigentliche Filename. Diese Anzeigen werden ständig aktualisiert, wenn Sie sich im oberen Bereich durch den Verzeichnisbaum klicken. Sie

können aber auch gleich die entsprechenden Angaben über die Tastatur eingeben. Mittels „OK" bestätigen Sie Ihre Wahl, was natürlich auch mittels Doppelklick auf den Filenamen geht, oder brechen Sie mit „Cancel" ab. Außerdem läßt sich über „Parent" ins darüberliegende Verzeichnis wechseln oder mit „Drives" eine Liste logischer Laufwerke anzeigen. Das sind zum einen Disketten- oder Festplattenlaufwerke, aber auch fest zugewiesene Verzeichnisse wie „S:" oder „DEVS:". Sollten Sie ein Freund der Menütechnik sein, so werden auch Sie bedient, denn all diese Funktionen lassen sich auch über ein Menü erreichen.
Doch nun zum „More"-Utility selbst. Dummerweise wurde immer noch keine angenehme Benutzeroberfläche eingebaut, so daß Sie hier voll auf Ihre Tastatur angewiesen sind. Alle Tastaturkürzel an dieser Stelle aufzuführen, würde jedoch den Absichten dieses Buches trotzen, weshalb nur die drei allerwichtigsten erwähnt seien. Mit der Return-Taste zeigen Sie die nächste Zeile an, mit der Space-Taste blättern Sie eine ganze Seite vor und mit der Backspace-Taste eine Seite zurück. Sollten Sie dieses Utility noch ausgiebiger nützen wollen, so seien Sie auf Ihr Handbuch verwiesen.

4.1.3 Display

Dieses kleine Utility zum Anzeigen von IFF-Grafiken sei nur der Vollständigkeit halber aufgeführt, denn da keine Verbesserungen an ihm vorgenommen wurden, ist es immer noch nur sehr unhandlich zu bedienen. Der beste Weg bleibt nach wie vor, ähnlich dem „More"-Utility zuerst das Icon des Bildes, das man anzeigen möchte

anzuwählen, und dann mit gehaltener Shift-Taste das „Display"-Icon doppelzuklicken. Da außerdem die wesentlichsten Funktionen in einem kleinen Fenster angezeigt werden, werden wir erst gar nicht Ihre kostbare Zeit verschwenden, sondern gleich fortfahren.

4.1.4 Say

Und auch das „Say"-Utility zur Ausgabe von Sprache ist im Grunde identisch mit dem der älteren Workbench und verdient keine weitere Beachtung. Noch dazu werden Sie es wohl niemals sinnvoll einsetzen können, weshalb es auch völlig sinnlos wäre, sich die ohnehin unkomfortable Bedienung einzuprägen. Lediglich ein kleiner Hinweis, falls Sie es doch einmal ausprobieren möchten und nichts funktioniert: Selbstverständlich benötigt dieses Programm den „Speak"-Treiber, der deshalb bei der Workbench 2.1 im „DOSDrivers"-Schub des „Devs"-Verzeichnisses stehen muß.

4.1.5 Exchange

Dieses Utility bedarf schon ein wenig mehr Beachtung. Allerdings gehört es nicht ins „Utility"-Verzeichnis und ist bei der Workbench 2.1 auch daraus entfernt worden. Denn

eigentlich ist es ein „Commodity" und gehört sich auch in den entsprechenden Schub. Da an dieser Stelle die Commodities sowieso noch nicht besprochen wurden, wird auch dieses Utility erst später, im entsprechenden Rahmen behandelt werden (siehe: 4.3.1 Exchange).

4.2 Workbench Tools

Einige durchaus brauchbare Hilfsprogramme sind unter dem Name „Tools" zusammengefaßt. Die meisten waren auch schon unter der Workbench 1.2 verfügbar, doch wurden Sie überarbeitet, was in den meisten Fällen nicht nur eine Anpassung an die neue Benutzeroberfläche bedeutet. Deshalb sollen auch diese Bestandteile der neuen Workbench wenigstens kurz angesprochen werden.

4.2.1 IconEdit

Der „IconEdit" ist das Tool, mit dem Sie die hübschen kleinen Bildchen auf der Workbench erstellen und verändern können. Denn sie sollen nicht nur das Auge erfreuen, sondern auch aussagekräftig sein, und wenn Sie statt eines Papierkorbs ein schwarzes Loch bevorzugen, wie Sie es von Ihrem Next gewohnt sind, dann ist das durchaus machbar. Doch was ist nun das Neue an dem Editor? Nun, da wären zum einen die besseren Editiermöglichkeiten, die über kleine Gadgets angewählt werden können. So können Linien gezogen oder Flächen gefüllt werden und darüber hinaus der letzte Vorgang mit der „Undo"-Funktion rückgängig gemacht werden. Weit interessanter als sie auf den ersten Blick erscheinen mögen, sind die „Normal"- / „Selected"-Knöpfe. Denn nun wird das erste Mal offiziell die Möglichkeit unterstützt, einem angewählten Icon ein völlig anderes Aussehen zu geben, als es im nichtangewählten Zustand hat. Und mit den beiden Knöpfen schalten Sie zwischen diesen beiden Zuständen um, können Sie also beide unmittelbar bearbeiten. Dazu müssen Sie allerdings zuvor definieren, daß es sich um solch ein solches Icon mit zwei Grafiken handelt. Dazu wählen Sie im „Highlight"-Menü die „Image"-Option an.

Überhaupt sind die Menüs in diesem Programm recht attraktiv. Neben der Wahl des Icon-Typs und der Hervorhebung bei der Anwahl sind gerade die letzten drei Menüs beachtenswert. So verbirgt sich im „Images"-Menü ganz unscheinbar hinter der „Load"-Option ein Untermenü, das auch das Laden von IFF-Brushes erlaubt. Mit anderen Worten, Sie können die Icons mit Deluxe Paint oder einem anderen Standard Malprogramm malen und anschließend im IconEdit ein Icon daraus machen, wodurch Sie natürlich weit komfortablere Bearbeitungsmöglichkeiten zur Verfügung haben, als ein solcher Editor je bieten kann. Der erste Menüpunkt aus dem „Extras"-Menü, „Recolor" ist gerade für den Umsteiger unbezahlbar. Denn mit dieser Funktion, lassen sich die Farbe zwei und drei vertauschen, was deshalb so wichtig ist, weil eben diese Vertauschung auch bei den Workbenchfarben durchgeführt wurde. Sollten Sie also bereits unter der Workbench 1.2 / 1.3 plastische Icons gezeichnet haben, so werden diese unter 2.0 alle eingedrückt dargestellt.

Ein kurzer Aufruf von „Recolor" genügt, und das Problem hat sich erledigt! Nicht neu, aber dennoch erwähnenswert ist die „Use Grid?"-Option im „Settings"-Menü. Mit Ihr läßt sich ein Netz über die Anzeige legen, wodurch die einzelnen Pixel besser abgrenzbar sind, bzw. wenn Sie diese Funktion abwählen, die tatsächliche Vergrößerung ohne irgendwelche irritierenden Spezialeffekte bekommen.

Außerdem versteht der „IconEdit" eine Vielzahl von „Tool Types", die jedoch nur in den seltensten Fällen sinnvoll anwendbar sind und im Grunde nur die Voreinstellungen für

den Editor definieren. Da es aber eher unwahrscheinlich ist, daß Sie genau eine dieser Funktionen unbedingt benötigen und die Besprechung doch recht lang wäre, soll in diesem Rahmen darauf verzichtet werden. Sie finden alle Parameter im Systemhandbuch umfassend erklärt.

4.2.2 MEmacs

MicroEMACS ist auch schon ein alter Bekannter und kann immer noch nicht mit professionellen Texteditoren mithalten. Allerdings bekam er wenigstens inzwischen umfangreiche Menüs verpaßt, so daß sich sogar mit ihm arbeiten läßt. Die Funktionsvielfalt ist sogar schon so überwältigend, daß auf dieses Tool im Rahmen der Blitzeinstiegs unmöglich eingegangen werden kann. Da aber später noch ein Texteditor gebraucht werden wird, werden wir uns noch dem „Ed" zuwenden, der mehr Shell-orientiert ist. Zu mehr als zum bloßen Texte editieren sind beide Editoren ohnehin ungeeignet.

4.2.3 GraphicDump

An dieses Programm hat sich merklich überhaupt nichts verändert. Zum Ausdruck des momentanen Bildschirminhalts genügt ein einfacher Doppelklick auf das „GraphicDump"-Icon. Danach bleiben Ihnen etwa zehn Sekunden (die Bombe tickt), um den Screen, den Sie ausdrucken möchten, in den Vordergrund zu bringen, Menüleisten herunterzuklappen oder Windows zu manipulieren. Das ist insbesondere wichtig, da beim Ausdruck der Workbench sonst immer das „GraphicDump"-Icon zu sehen wäre. So bleibt Ihnen Zeit, das „Tools"-Window zu schließen. Beachten Sie

auch, daß der richtige Drucker und die richtigen Grafikdruck-Parameter eingestellt sind (siehe hierzu: 3.1.9 Printer und 3.1.10 PrinterGfx).
Wer noch Ansprüche an die Größe des Ausdrucks stellt, der kann dem Programm seine Wünsche über die „Tool Types" vermitteln. Dazu wählen Sie das Icon von „Graphic Dump" an und gehen anschließend auf „Info" im „Icons"-Menü. Dort können Sie unter „Tool Types" die gewünschte Größe angeben. Zur Auswahl stehen: „tiny" (winzig = 1/4 der maximalen Größe), „small" (klein = 1/2), „medium" (mittel = 3/4) und „large" (groß = 1/1). Sie können aber auch direkt die Größe in Punkten angeben, und zwar im Format „xdots:ydots" (horizontale Punkte:vertikale Punkte). Der gewünschte Parameter wird hinter ein „size=" geschrieben und das war dann auch schon alles. Beispiele wären: „size=medium" oder „size=200:100".

4.2.4 Colors

Wenn Sie sich jetzt vielleicht fragen werden: „Was, schon wieder ein Programm zur Einstellung der Bildschirmfarben - dafür gibt doch schon 'Palette' in den Prefs, was soll das?", dann muß man Ihnen auf den ersten Blick recht geben, denn auch mit „Colors" läßt sich - ebenso wie mit „Palette" - die Farbtabelle eines Bildschirms verändern. Aber eben „eines" Bildschirms, und zwar eines beliebigen und nicht nur die des Workbench-Screens. Denn „Colors" kann sein Fenster auf jedem beliebigen Screen öffnen - was auch ein neues Feature von AmigaOS 2.0 ist. Doch wahrscheinlich ist hierfür ein Beispiel angebracht.

Starten Sie „MEmacs". Dieser wird seinen eigenen Screen öffnen, den Sie nun ein gutes Stück nach unten ziehen, so daß die Workbench wieder sichtbar wird. Dort

Doppelklicken Sie nun „Colors" und dieses öffnet sein Window nun nicht auf der Workbench, sondern auf dem MicroEmacs-Screen, oder allgemein gesagt auf dem vordersten. Jetzt können Sie den Screen wieder hochziehen, im „Colors"-Window die Farben nach Ihren Vorstellungen editieren und das Programm wieder verlassen. Der Vorteil liegt auf der Hand: Auch wenn ein Programm nicht die Möglichkeit bietet, seine Farben zu verändern, so genügt ein Doppelklick und Sie haben keine Probleme mehr. Andererseits braucht ein Programmierer nicht unbedingt seinen eigenen Palette-Requester einbauen, obwohl das natürlich immer noch die komfortabelste Möglichkeit bleibt und das „Colors"-Tool nur ein Hilfsmittel für den Notfall sein kann.

4.2.5 Calculator

Der „Calculator" macht von dieser neuen Funktion leider keinen Gebrauch, obwohl gerade ein Taschenrechner oftmals bei der Nutzung anderer Programme dringend gebraucht werden könnte. Doch vielleicht kommt dieser Gedanke den Entwicklern von Commodore auch noch, warten wir auf die Workbench 2.2. Bis dahin gibt es in Sachen Taschenrechner nichts neues zu melden: man kann halt mit ihm rechnen, auch ganz nett.

4.2.6 InitPrinter

Je nachdem welchen Drucker Sie besitzen und in welcher Form Sie Ihren Ausdruck bevorzugen, stellen Sie in den Printer-Preferences Ihre Wünsche ein. Und diese Daten werden vor dem ersten Ansprechen des Druckers an diesen übermittelt. Wenn Sie jedoch nach diesem ersten Ausdruck den Drucker zwischendurch abschalten und

anschließend wieder mit Ihren gewohnten Einstellungen arbeiten wollen, so müssen diese Daten nochmals übertragen werden. Und genau das passiert, wenn Sie das Programm „InitPrinter" starten.
Diese Verfahrensweise ist weit flexibler, als wenn die Daten beispielsweise vor jedem Ausdruck erneut gesetzt werden würden, da Sie ja vielleicht zwischendurch auf andere Einstellungen umschalten möchten, z. B. mit einem Schalter am Drucker. Denn dann könnten derartige Eingriffe „von außen" überhaupt keinen Einfluß nehmen. Mit dem „InitPrinter"-Programm liegt es in Ihren Händen, wann der Drucker zurückgesetzt wird.
Solange Sie den Drucker eingeschaltet lassen und kein anderes Anwendungsprogramm die Konfigurationen ändert, brauchen Sie selbstverständlich nicht jedesmal „InitPrinter" aufrufen, da die Daten für diesen Zeitraum im Drucker gespeichert sind.

4.2.7 KeyShow

Das „KeyShow"-Programm ist nicht für den täglichen Gebrauch geschaffen, sondern dient mehr der Information: Es zeigt die momentane Tastaturbelegung an. Dazu wird ein Abbild Ihrer Tastatur auf dem Monitor angezeigt und alle Tasten mit den ihnen zugeordneten Buchstaben dargestellt. Durch Klick auf die Ctrl- oder eine der Shift- oder Alt-Tasten können Sie sich zudem anzeigen lassen, welche Zeichen mit den entsprechenden Umschalttasten erscheinen. Dabei ist das Programm so intelligent, daß das Aussehen der Tastatur abhängig von den verwendeten Einstellungen richtig gewählt wird (die amerikanische Tastatur hat eine größere Return-Taste, dafür fehlt die Taste, die bei der deutschen Tastatur mit dem „#" belegt ist). Wenn die Taste einen Steuercode enthält, so beginnt die Anzeige mit „~" oder „^". Sollte mehr als nur ein

einziger Buchstabe auf einer Taste liegen (was durchaus möglich ist), so weist ein „$$" darauf hin.
Daß dieses Programm aber durchaus seine Berechtigung hat und nicht nur eine bloße Spielerei ist, macht vielleicht folgendes Beispiel deutlich: Gerade in den Anfangszeiten des Amigas, als es nur recht vereinzelte Amiga-Anwender gab, und die Kommunikation deshalb sehr eingeschränkt war, wurde selbst in Fachzeitschriften mehrfach die Frage behandelt, wo denn der Apostroph auf der Tastatur zu finden sei. Denn das Ding, das sich ganz links oben, unterhalb der Esc-Taste befindet, ist leider spiegelverkehrt. Und gerade für die damals noch scharenweise vorhandenen AmigaBASIC-Anwender war diese Taste sehr wichtig, denn Sie entspricht dem „REM"-Befehl - und abgetippte Programme, in denen die falsche Taste verwendet wurde, liefen einfach nicht. Mit dem „KeyShow"-Programm wäre das alles kein Problem gewesen, Sie hätten nach kurzer Zeit den Apostroph auf der Kombination Alt-ä orten können.
Und auch wenn gerade dieses eine Zeichen inzwischen hinreichend bekannt sein dürfte, so sucht man doch immer wieder nach irgendwelchen Exoten, wie etwa „3/4". Und da ist „KeyShow" dann schon eine enorme Hilfe.

4.2.8 PrintFiles

Dieses Programm zählt zweifellos zu einem der wirklich unentbehrlichen Utilities für den Workbench-Anwender. Wann immer Sie ein oder mehrere Textfiles ausdrucken wollen, klicken Sie diese(s) einfach an und Doppelklicken anschließend bei gedrückter

Shift-Taste „PrintFiles". Zur Auswahl mehrere Files benützen Sie die erweiterte Auswahl (entweder die Shift-Taste oder den Kasten, siehe hierzu: 2.3 Menüs).
Da es oftmals sehr wünschenswert ist, zwischen zwei aufeinanderfolgenden Texten einen Seitenvorschub einzufügen, können Sie in den „Tool Types" zu „PrintFiles" den Parameter „FLAGS=fromfeed" setzen, der genau das bewirkt.

4.2.9 ShowConfig

Dieses Programm gibt Ihnen die Konfigurationsdaten Ihres Rechners aus. Dazu zählen der Prozessor, die Custom-Chips, die Versionen der Kickstart, der RAM-Speicher sowie die eingebauten Erweiterungskarten. Da Sie wohl meistens wissen, welche Karten Sie in Ihren Rechner gesteckt haben und die Informationen hierzu sehr spärlich sind, helfen diese nicht sonderlich weiter. Wesentlich interessanter sind da schon die Custom-Chip Revisionen, denn nun brauchen Sie nicht extra den Rechner aufzuschrauben, nur um herauszufinden, ob Sie nun schon eine ECS-Denise haben oder nicht. Und da sowieso nicht jeder mit den auf den Chips aufgedruckten Nummern etwas anfangen kann, ist diese Anzeige höchst willkommen. Bevor Sie nun also in den nächsten Laden laufen, um sich einen neuen Chip zu leisten, starten Sie dieses Programm und überprüfen Sie, ob das Teil nicht vielleicht schon in Ihrem Rechner steckt!

4.2.10 CMD

Dieses kleine Tool ermöglicht es Ihnen, die Ausgabe von Daten auf eine Ausgabeeinheit wie beispielsweise den Drucker in ein File umzulenken. Dies kann sinnvoll sein,

wenn Sie einen Text entsprechend Ihrem Druckformat mit dem Seitenvorschub versehen möchten ohne es auszudrucken. Dann wählen Sie einfach „PrintFiles" an und lassen dieses die nötigen Steuercodes in den Text einfügen und das Ergebnis in ein neues Textfile fließen. Ein andere Anwendung besteht in der Fehlersuche; falls irgendein Ausdruck nicht so aussieht, wie er soll, kann es hilfreich sein, zu überprüfen, welche Daten an den Drucker geschickt wurden, um gegebenenfalls den Druckertreiber zu ändern. Dazu müssen Sie aber erst einmal an die Daten herankommen, und das geht mit „CMD".
Welchen Port Sie umlenken, und wohin die Daten dann gehen sollen, können Sie in den „Tool Types" von „CMD" einstellen. Unter „DEVICE=" können Sie hier „parallel" oder „serial" angeben, je nachdem, wo das Gerät angeschlossen ist, das Sie umgehen wollen. Dann geben Sie noch bei „FILE=" Pfad und Name des Files, in das die Ausgabe gehen soll an und starten „CMD". Die nächste Ausgabe auf die entsprechende Einheit geht nun in das angegebene File. Wenn Sie nicht nur eine einzige Ausgabe umlenken wollen, so sollten Sie zudem „MULTIPLE=true" setzen, denn dann werden alle folgenden Ausgaben in Files mit den Namen „Name.Nummer" gemacht. Wenn Sie also „FILE=ram:CMD_file" gewählt haben, so gehen die Daten in „ram:DMD_file.1", „ram:CMD_file.2", usw. Um diese Umlenkung wieder aufzuheben, genügt ein erneuter Doppelklick auf „CMD".

4.2.11 HDBackup & HDToolBox

Diese beiden Harddisk-Utilities sind eigentlich keine Workbench-Tools an sich, sondern eher Beigaben für die Harddisk-Benutzer. Deshalb gehören beide Programme eigentlich nicht in einem Blitzeinstieg zu AmigaOS 2.0. Vor allem, deshalb weil das

Programm „HDBackup“ umfangreich genug ist, um ein eigenes Buch zu füllen.

4.3 Commodities

Die „Commodities“ sind ebenfalls im „Tools“-Verzeichnis anzutreffen, haben aber ihre eigene Schublade und sind auch vom Prinzip her etwas ganz anderes als die Tools. Was ein Commodity erst zum Commodity macht, ist die Eigenschaft, daß es sich in den Input-Handler hängt und somit alle möglichen Eingaben schon vor dem Rest des Systems bekommt. Diese kann es dann nach belieben weiterverarbeiten oder auch (modifiziert) weiterleiten.
Sie können beliebig viele Commodities starten, die dann alle im Hintergrund laufen. Doch bedenken Sie immer, daß abhängig davon, was das Programm macht und wie gut es programmiert wurde es einen mehr oder weniger ins Gewicht fallenden Anteil der Rechnerleistung verbraucht. Mit anderen Worten: Man kann alles übertreiben, auch die Nutzung von Commodities. Überlegen Sie sich, welche Sie wirklich brauchen und verzichten Sie auf die restlichen.
Da ja alle Commodities im Input-Handler hängen, und nacheinander die einlaufenden Daten untersuchen und weiterreichen, ist es klar, daß ihre Reihenfolge äußerst entscheidend ist. Stellen Sie sich vor, zwei Commodities warten auf eine ganz bestimmte Tastenkombination, die sie anschließend ausfiltern. Das zweite Commodity würde bis in alle Ewigkeit warten! Deshalb kann man bei allen Commodities in den „Tool Types“ unter „CX_PRIORITY“ die Priorität bestimmen. Dadurch kann dann eine eindeutige Ordnung erstellt werden.

4.3.1 Exchange

Das Programm „Exchange" ist das Steuerungsprogramm der Commodities. Sie können diese zwar nur durch einen Doppelklick starten, dann jedoch werden sie in die Liste verfügbarer Commodities aufgenommen und Sie können sie weiter bearbeiten. Exchange öffnet ein eigenes Window, indem zusätzliche Informationen zum angewählten Commidity angezeigt werden. Wenn dieses auch ein eigenes Window zur Verfügung stellt (um beispielsweise weiter Parameter anzugeben), so kann es geöffnet bzw. geschlossen werden. Außerdem kann es vorübergehend inaktiv geschaltet, oder gar ganz entfernt werden, was natürlich auch durch erneuten Doppelklick auf das Commodity-Icon selbst geschehen kann.

4.3.2 AutoPoint

Da Sie ja sicher den standardmäßigen „AutoPoint" von Ihrer SUN-Station vermissen, ist dieses Tool genau das, was Sie immer vermißt haben. Doch im Ernst: Es gibt Rechner (eben z.B. die SUN), bei denen es genügt, den Mauszeiger über ein Window zu bewegen, um es zu aktivieren. Beim Amiga wird ein Window ja üblicherweise erst aktiviert, nachdem in sein Inneres geklickt wurde. Mit „AutoPoint" schalten Sie die alternative Verfahrensweise ein. Hier ist auch sehr gut das Vorgehen eines Commodities zu erkennen: Es fängt die Mausbewegung ab, und wenn sich der Mauszeiger über einem neuen Window befindet, wird dieses aktiviert. Natürlich werden die Mausdaten ebenso wie alle anderen anschließend weitergeleitet.

4.3.3 ClickToFront

Ebenfalls etwas mit Windows hat „ClickToFront" zu tun. Gerade bei der intensiven Nutzung vieler Windows gleichzeitig kommt es oftmals vor, daß man ein Fenster im Vordergrund haben will, dessen Vorder-/Hintergrund-Symbol unglücklicherweise gerade überlagert ist. Dann hilft „ClickToFront". Denn wenn es gestartet wurde, genügt ein Doppelklick irgendwo in das Fenster (nur nicht gerade auf ein Icon) um dieses nach vorne zu holen.
Falls Sie diese Funktion jetzt gleich gierig ausprobieren mußten und nichts funktioniert hat, werfen Sie nicht gleich den guten Blitzeinstieg in die Ecke, sondern schauen Sie sich einmal die „Tool Types" von „ClickToFront" an: Da steht u.a. „QUALIFIER=irgendwas". Und wenn dieses „irgendwas" nicht gerade „NONE" ist, so bedeutet das, daß Sie zu dem Doppelklick auch noch eine bestimmte Taste gedrückt halten müssen. Zur Auswahl stehen hierbei: LEFT_ALT, RIGHT_ALT und CTRL. Eine dieser Tasten zusätzlich zum Doppelklick zur Bedingung zu machen, ist gar keine so schlechte Idee, dann holen Sie wenigstens kein Window unbeabsichtigt nach vorne. Andererseits bedeutet das natürlich auch wieder ein gewisses Maß an Mehraufwand - wofür Sie sich letztlich entscheiden, ist Geschmacksache.

4.3.4 Blanker

Sicherlich das Commodity überhaupt ist der „Blanker". Wie Sie ja sicher wissen, sind Monitore recht empfindliche Geräte, die es auf den Tod nicht ausstehen können, wenn über längeren Zeitraum eine helle Grafik dargestellt werden soll. Diese kann nämlich in die Beschichtung des Bildschirms, die ja unablässig mit Elektronen beschossen wird, einbrennen. Um dem vorzubeugen, sollte das Bild bei längerer Nichtbenutzung dunkel geschaltet werden, z.B. wenn Sie sich mal eben eine Tasse Kaffee holen. Den

Monitor auszuschalten wäre auch falsch, denn häufiges An- und Ausschalten verringert die Lebensdauer eines jeden elektronischen Geräts.

Und eben deshalb ist der „Blanker" das ideale Commodity. Er erlaubt es, einen Zeitraum vorzugeben, nach dessen Ablauf der Bildschirm schwarz geschaltet wird, wenn inzwischen keine Taste betätigt, bzw. die Maus bewegt wurde. Denn auf diese Weise erkennt der Rechner automatisch, daß Sie ihn im Moment nicht benötigen, und kann entsprechend reagieren.
Bei der Workbench 2.1 wurde der Blanker sogar noch etwas erweitert: Nun können Sie optional auch noch farbige Linien über den Bildschirm tanzen lassen, solange der Bildschirm schwarz ist, oder aber den kompletten Bildschirm farbig pulsieren lassen. Das zeigt Ihnen zum einen an, daß der Monitor noch nicht ausgeschaltet ist und zum anderen ist, es ganz einfach nett anzusehen.

4.3.5 NoCapsLock

Bei diesem Commodity geht es um die Tastatureingabe. Es bewirkt, daß die Caps-Lock Taste (das ist die mit dem LED), die normalerweise auf Großschreibung schaltet, inaktiv wird. Die Rechtfertigung für dieses Programm ist, daß Sie nicht mehr „aus Versehen" die Caps-Lock Taste betätigen, und im folgenden alles groß schreiben können. Da man aber ebenso jede andere Taste aus Versehen betätigen kann, und dieser Fehler wohl weniger auffällt als eine komplett groß geschriebene Textzeile, ist dieses Tool wohl durchaus verzichtbar.

4.3.6 IHelp (WB 2.0)

„IHelp" gibt der Tastatur hingegen zusätzliche Funktionen, anstatt bestehende zu deaktivieren. Mit diesem Programm lassen sich nämlich alle möglichen Funktionen, die normalerweise mittels der Maus angewählt werden, auf die Tastatur legen. Damit wird es wieder einmal noch einfacher, den Amiga nur über Tasten zu bedienen. Zwar ist die Maus ein phantastisches Hilfsmittel und einer der Hauptgründe, warum das Arbeiten mit dem Amiga so intuitiv vonstatten gehen kann, wenn man jedoch ohnehin ständig Tastatureingaben machen muß, kann der Wechsel zwischen den beiden Eingabemedien doch lästig werden.
Doch wie funktioniert das nun? Zunächst einmal muß darauf hingewiesen werden, daß diese Commodity unter der Workbench 2.1 nicht mehr existiert, da seine Funktion in dem neuen „FKey" integriert wurde (siehe: 4.3.8 FKey (WB 2.1)). Als Workbench 2.0 Benutzer muß man wieder einmal den Umweg über die „Tool Types" einschlagen. Dort sind eine Reihe von Funktionen aufgeführt, wie etwa „CYCLE=" oder „MAKEBIG=", die Sie auf eine Taste legen können, indem Sie deren Bezeichnung hinter das „=" setzen. Welche Kombinationen für die Wahl der Tasten möglich sind, entnehmen Sie bitte dem Kapitel: 4.3.9 Tastenkombinationen. Die einzelnen Schlüsselwörter bedeuten dabei folgendes:

CYCLE
Blättern in Anwendungswindows

MAKEBIG
aktives Window auf maximale Größe vergrößern

MAKESMALL
aktives Window auf minimale Größe verkleinern

CYCLESCREEN
Bildschirme blättern
ZIPWINDOW
aktives Window zoomen (entspricht Zoom-Gadget)

4.3.7 FKey (WB 2.0)

Auch „FKey" belegt die F-Tasten (was auch der Grund ist, warum bei der Workbench 2.1 für beide Funktionen nur noch ein Programm existiert). Es erlaubt die Belegung der F-Tasten mit Texten. Wenn Sie es starten, wird ein übersichtliches Window geöffnet, in dem Sie zu jeder F-Taste einen beliebigen Text eingeben können. Am unteren Ende ist noch ein mit „Modifier" betiteltes Gadget, bei dem Sie noch auswählen können, ob zusätzlich zur F-Taste auch noch die Shift-Taste gehalten werden muß, so daß es insgesamt möglich ist, 20 Tasten zu gelegen (mit Shift und ohne Shift). Das ganze läßt sich dann noch abspeichern um die Eingaben dauerhaft zu machen.

4.3.8 FKey (WB 2.1)

Wie bereits erwähnt sind die Funktionen des „IHelp" Commodity in das neue „FKey" integriert worden. Von nun an können sowohl Texte als auch Funktionen auf jede beliebige Taste gelegt werden. Auch die umständliche Handhabung mit den „Tool

Types" entfällt. Es wird ein sauberes Window geöffnet, in dem Sie eine Liste der belegten Tasten angezeigt bekommen. Mit „Taste dazu" fügen Sie eine neue Taste hinzu, mit „Taste löschen" entfernen Sie eine vorher definierte. Nachdem Sie die Bezeichnung der Taste eingetippt haben (siehe: 4.3.9 Tastenkombinationen) können Sie mit dem unter „Befehl" stehenden Gadget die Funktion der Taste auswählen. Dabei stehen die bereits unter „IHelp" erwähnten Funktionen zur Verfügung, ebenso wie „Text einfügen", was dem bisherigen „FKey" der Workbench 2.0 entspricht und zwei neue Varianten: „Programm starten" und „ARexx-Skript starten". In beiden Fällen gehört der zugehörige Filename in das Text-Gadget „Befehlsargumente". Auf diese Weise können Sie ganze Programme mit nur einer einzigen Taste starten - die schnellste Möglichkeit überhaupt!

4.3.9 Tastenkombinationen

Um die letzten Programme auch sinnvoll anwenden zu können, ist es nötig zu wissen, wie man die Tasten eigentlich korrekt bezeichnet. Dabei soll Ihnen folgende Tabelle eine Hilfe sein:

Alt
eine der beiden Alt-Tasten

RAlt
die rechte Alt-Taste

LAlt
die linke Alt-Taste

Shift
eine der beiden Shift-Tasten

RShift
die rechte Shift-Taste

LShift
die linke Shift-Taste

LCommand
die linke Amiga-Taste

RCommand
die rechte Amiga-Taste

Control
dir Ctrl-Taste

Numericpad
eine Taste vom Nummernblock

Rbutton
der rechte Mausknopf

Leftbutton
der linke Mausknopf

Diese Tasten können natürlich auch miteinander kombiniert werden und meistens kommt auch noch ein Buchstabe, eine Ziffer oder F-Taste hinzu. Beispiele hierfür sind: „LAlt b", „Control Shift F1" oder „Leftbutton x".

4.3.10 CrossDOS (WB 2.1)

Zu den neuen Treibern zum Ansprechen der Laufwerke im PC-Format (PC0:, PC1:, ...) gehört auch ein Commodity. Da der ASCII keineswegs ein echter „Standardcode" ist, wie er es von sich behauptet, sondern immer noch viel Platz für eigene Definitionen läßt, hat natürlich auch jede Firma ihre eigene Variante. Das fällt im normalen Betrieb nicht weiter auf, da die ersten Codes alle noch wohldefiniert sind, doch bereits bei Umlauten merkt man eben doch, daß die Definitionen nicht vollständig sind. Daher ist ein „ä" auf dem Amiga kein „ä" auf dem PC und umgekehrt. Dazu kommt dann noch die Eigenheit von PCs, nicht nur ein Linefeed ans Zeilenende anzuhängen, wie es beim Amiga der fall ist, sondern auch noch ein Carriage Return. Deshalb sehen Texte, die man 1:1 vom PC übernommen hat, immer etwas seltsam aus.

Um eben diesen Effekt zu umgehen, kann man Textfilter und Textkonvertierung einschalten, die sich selbstverständlich wirklich nur auf Texte bezieht, Programme können Sie auf einem normalen Amiga ohnehin nicht starten. Außerdem läßt sich noch die Konvertierungsart bestimmen und das ganze jeweils getrennt für jedes einzelne Laufwerk.

4.4 System

Auch im „System"-Ordner sind noch einige Utilities enthalten, die wohl nicht so ganz in den „Utilities"-Ordner gepaßt haben. Dennoch sind es Utilities, die Sie ebenso gebrauchen können, wie alle anderen.

4.4.1 NoFastMem

„NoFastMem" begleitet den Amiga nun auch schon eine Zeit lang. Falls Sie immer noch nicht wissen, was es eigentlich tut: Es belegt den gesamten Fast-Mem Speicher (und gibt ihn bei einem erneuten Aufruf wieder frei). Nötig ist dies eigentlich nur bei sehr alten und unsauber geschriebenen Programmen, die davon ausgehen, daß nur Chip-Mem vorhanden ist. Da solche Programme aber einerseits schon längst durch neuere und bessere Versionen ersetzt worden sein sollten und andererseits ein Programm, daß nicht einmal mit Fast-Mem zurecht kommt, wohl kaum die Kickstart 2.0 verkraftet, ist die Anwendung von „NoFastMem" doch recht zweifelhaft. Wenn Sie also nicht zufälligerweise ein unverzichtbares Programm besitzen, daß nur nach dem Start von „NoFastMem" funktioniert, dann vergessen Sie es!

4.4.2 Setmap (WB 2.0)

Nur noch auf der Workbench 2.0 vorhanden ist „Setmap" - bei der Workbench 2.1 gehört die Tastaturbelegung ja bereits zu den Voreinstellern und wird dementsprechend mit einem Preferences-Utility gesetzt. Da die richtige Tastaturbelegung sowieso beim Systemstart von der „Startup-Sequence" geladen werden sollte, ist dieses Programm nur sehr selten nötig. Sollten Sie aber dennoch zwischendurch auf eine andere Tastaturbelegung umschalten wollen, so geben Sie in den „Tool Types" „KEYMAP=" und die Länderkennung an. Zur Auswahl stehen hier normalerweise „d" (deutsch), „f" (französisch), die standardmäßige „usa" und andere.

4.4.3 DiskCopy (WB 2.1)

Das Programm „DiskCopy" wird auch nur noch unter der Workbench 2.0 über ein Icon aufgerufen. Da diese Funktion bei der neuen WB 2.1 in den „Copy"-Befehl aus dem „Icons"-Menü integriert wurde, sollten Sie sich wenn nötig den entsprechenden Abschnitt unter: 2.3.3 Icons durchlesen.

4.4.4 Format

Auch „Format" ist ja nichts neues, aber es ist dennoch nach wie vor notwendig. Mit ihm formatiert man Disketten oder andere Speichermedien und macht Sie somit nutzbar. Unter der Workbench 2.0 muß nach wie vor erst ein Diskettensymbol angeklickt werden, bevor man durch Doppelklick auf das Format-Icon (unter Halten der Shift-Taste versteht sich) den Formatiervorgang starten kann.

Die neue Version, die zur Workbench 2.1 gehört, zeichnet sich hingegen durch die aufgepeppte Benutzeroberfläche auf, die nun die Auswahl aus der Liste möglicher Devices zuläßt. Dabei erhält man zusätzlich Informationen über die Kapazität des Mediums und zu welchem Anteil es belegt ist. Nachdem man das gewünschte Device angewählt hat, können noch weitere Angaben zur Formatierung selbst gemacht werden. Da ist natürlich zuerst einmal der Name der Diskette, der nun nicht mehr nachträglich mittels eines „Rename"-Aufrufs umbenannt werden muß. Darüber hinaus läßt sich bestimmen, ob ein Trashcan eingerichtet und ob das FastFileSystem benutzt werden soll. In letzterem Fall ist es nicht mehr möglich, auf die Diskette mit einem Betriebssystem vor 2.0 zuzugreifen. Daran sollten Sie denken, wenn Sie die Diskette weiterreichen möchten. Andrerseits passen auf eine FFS-Diskette mehr Daten und der Zugriff erfolgt auch schneller. Gestartet wird die eigentliche Formatierung dann mittels Klick auf „Formatieren" für eine volle Formatierung, oder auf „Schnell" für die schnelle Formatierung von Disketten, die schon einmal im AmigaDOS-Format beschrieben wurden.

4.4.5 RexxMast

Ein weiteres Geheimnis der Workbench 2.0 verbirgt sich hinter dem Programm „RexxMast". Da der Begriff „ARexx" inzwischen schon mehrfach durch alle Zeitschriften gegangen ist und wohl jeder nur halbwegs interessierte Anwender zumindest eine grobe Vorstellung davon hat, worum es dabei geht, traut man sich kaum, es noch einmal auszusprechen: ARexx ist eine Programmiersprache, die in besonderem Maße die Interaktion von Programmen unterstützt. Soweit so gut. Also praktisch eine

Art BASIC, das von mehreren Anwendungen angesprochen werden kann. Nun, da in diesem Buch absichtlich das Thema ARexx fast völlig ignoriert wird (dafür gibt es genügend Bücher von ganz anderem Umfang!), machen wir es an dieser Stelle auch ganz kurz: Egal was ARexx nun letztendlich wirklich ist und was genau man damit anfangen kann, wesentlich ist, daß Sie, bevor Sie darauf zugreifen können, zuerst einmal das Programm „RexxMast" starten müssen. Es läuft dann unscheinbar im Hintergrund und springt erst dann an, wenn ARexx wirklich genutzt wird. Ohne dem „RexxMast" jedoch läuft gar nichts.
Wer also häufiger mit ARexx arbeiten möchte, der sollte den „RexxMast" auf jeden Fall gleich beim Start der Workbench automatisch aktivieren lassen. Wenn Sie allerdings nur selten von den Qualitäten von ARexx Gebrauch machen wollen, genügt es, das Programm über sein Gadget im System-Ordner zu starten. So sparen Sie ein klein wenig Speicher, da das Programm ja resident gehalten werden muß.

4.4.6 FixFonts

Um zu verstehen, was „FixFonts" eigentlich macht, muß ein klein wenig auf die Struktur der Amiga Font-Daten eingegangen werden. Zunächst einmal liegen Fonts immer im „Fonts:"-Verzeichnis. Dort existiert zu jedem Font eine Datei mit dem Namen des Fonts plus der Endung „.font" und ein eigenes Directory, das ebenfalls den Namen des Fonts trägt und in dem die eigentlichen Daten gespeichert werden. Diese sind dann in Files gespeichert, die die Höhe des Zeichensatzes als Name tragen. Um ein Beispiel zu nennen: Der Opal-Font hat ein File „Opal.font" und ein Verzeichnis „Opal" mit den Files „9" und „12". Das File „Opal.font" enthält dabei einige Daten die den Font betreffen und auch noch die zur Verfügung stehenden Höhen, im Beispiel also die Werte „9" und „12".

Was passiert nun, wenn Sie das File „9" löschen? Das File ist dann weg, die „9" steht aber immer noch im „Opal.font". Umgekehrt stimmen die Daten auch nicht mehr überein, wenn nachträglich eine neue Höhe hinzukopiert wurde. Und um genau diese Falschinformationen in den „.font"-Files zu korrigieren, gibt es das Programm „FixFonts". Es durchsucht das komplette „Fonts"-Verzeichnis und seine Unterverzeichnisse und ändert die Einträge in den „.font"-Dateien entsprechend um. Sollten Sie also jemals Änderungen an Ihrem Font-Directory durchführen, rufen Sie einfach hinterher „FixFonts" auf und alles ist wieder in Ordnung.

4.4.7 Fountain

Ebenfalls mit Fonts hat das Programm „Fountain" zu tun. Neben der oben erwähnten standardmäßigen Vorgehensweise der Bitmap-Fonts mit fest definierten Höhen wird seit der Workbench 2.0 auch das Konzept der Vektorfonts unterstützt. Vektorfonts sind Zeichensätze, die nicht als Grafikelemente abgespeichert sind, sondern als Berechnungsvorschriften. Da heißt es dann z.B.: Mache einen geraden Strich und verbinde die beiden Enden mit einem Bogen auf der rechten Seite und du erhältst ein großes „D". Und derartige Informationen können dann natürlich unabhängig von der Größe in gleichbleibend guter Qualität dargestellt werden, was gerade in Verbindung mit der Druckerausgabe eine große Rolle spielt, da die Auflösung von Druckern im Normalfall höher ist als die von Monitoren, weshalb dort die Buchstaben größer sein müssen.
Doch jetzt zu „Fountain" selbst: Alle Gadgets ausführlich zu beschreiben, wäre an dieser Stelle wohl ein wenig zuviel des Guten, denn die Bedienung ist auch so recht einleuchtend, wenn man weiß, was man mit dem Programm anfangen kann. Bevor Sie

Fonts in beliebiger Größe benützen, müssen Sie zuerst berechnet werden, und diesen Teil übernimmt „Fountain". Dabei geben Sie einfach den gewünschten Font und dann die Größe an, die Sie gerne hätten. Außerdem ist er möglich, in der gewählten Größe auch einen Bitmap-Font berechnen zu lassen, was die Textausgabe erheblich beschleunigt, da ja nun keine weiteren Berechnungen bei der Anzeige selbst mehr nötig sind. Andererseits kostet das natürlich wieder Speicherplatz auf der Diskette. Als Kompromiß sollten Sie sehr häufig verwendete Größen zu Bitmap-Fonts vorberechnen lassen, weniger gebräuchliche nur definieren.

4.4.8 CLI (WB 2.0) / Shell (WB 2.1)

Zweimal das gleiche unter einem anderen Namen ist die Schnittstelle zum „Command Line Interface" (CLI), das inzwischen eine „Shell" aufweist. Um die Begriffe zu klären: Prinzipiell wird das Fenster, das Sie mit einem netten kleinen Prompt zur Eingabe irgendwelcher Befehle auffordert, „CLI" genannt. Da aber im Zuge der Weiterentwicklung auch schon solche Feinheiten wie editierbare Zeilen hinzugekommen sind, wird das ganze jetzt unter der wohlklingenden Bezeichnung „Shell" angepriesen, was jedem Computerbesitzer das Gefühl von Sicherheit und Wohlstand auf Tastaturebene vermittelt.
Doch mal ganz im Ernst: Das „CLI" der Workbench 2.0 unterscheidet sich in keinster Weise vom „Shell" der Workbench 2.1, und selbst unter der Workbench 1.3 wurde gelegentlich schon der Begriff „Shell" verwendet. Was uns aber letztendlich interessiert, ist ja nichts weiter, als was wir mit dem Ding anfangen können. Und um diese Frage zu beantworten, müssen Sie sich vergegenwärtigen, daß Sie mit der „Shell" nicht eine Art von Utility starten, also nicht eine Ebene nach oben klettern, sondern

vielmehr eine Stufe hinabsteigen, zu der dem Rechner näher stehenden Kommando-Ebene.

Doch einem derart umfangreichen Thema wollen wir auch ein eigenes Kapitel widmen (6. Die Shell). Wesentlich ist im Moment nur, daß Sie diese ominöse Shell von der Workbench aus mit dem entsprechenden Icon im System-Ordner betreten.

5. Anwendung der Workbench

Wenn Sie die bisherigen Kapitel aufmerksam durchgearbeitet haben, sollten Sie jetzt wissen, wie man die Workbench konfiguriert und haben dies vielleicht auch schon getan. Die Bedienung jedes beliebigen Workbench-Utilities ist zum Kinderspiel geworden, Commodities sind keine außerirdischen Wesen mehr. Doch was noch fehlt, ist die eigentliche Anwendung der Workbench. Denn die Workbench existiert ja nicht zum Selbstzweck, sie soll die Plattform für das Arbeiten mit anderen Programmen darstellen. Und wie man das möglichst effektiv macht, werden Sie jetzt erfahren.

5.1 Installation von Anwendersoftware

5.1.1 WBStartup

Ein Ordner, der bisher noch nicht angesprochen wurde, und das, obwohl er sich im Hauptverzeichnis befindet, ist der „WBStartup"-Schub. Das ist auch nicht weiter verwunderlich, schließlich ist er ja leer. Füllen soll ihn der Anwender, also Sie, und zwar mit solchen Programmen, die Sie bei jedem Neustart der Workbench automatisch gestartet haben möchten. Damit entfällt das lästige Ändern der „Startup-Sequence", was unter WB 2.0 ohnehin verpönt ist.

Wenn Sie also ein Utility haben, das im Hintergrund laufen soll, dann ziehen Sie sein Icon einfach in dieses neue Verzeichnis. Ein sehr gutes Beispiel ist der „RexxMast". Sollten Sie öfters auf ARexx zurückgreifen wollen, so ist der „WBStartup"-Ordner das richtige Verzeichnis. Von nun an brauchen Sie nicht jedesmal von Hand den RexxMast aktivieren.

5.1.2 Expansion

Ein weiterer Ordner, der anfangs noch leer ist, und erst später gelegentlich Zuwachs erhält, ist der „Expansion"-Schub. Er ist zur Aufnahme von Treibern gedacht, wie sie für einige Erweiterungskarten benötigt werden. Abgesehen von Speichererweiterungen benötigen fast alle zusätzlichen Steckkarten spezielle Treiber, die die neue Hardware dann ansprechen. Diese können entweder in einem EPROM auf der Karte selbst enthalten sein, wie es für Festplatten-Adapter nötig ist, oder aber nachgeladen werden - und das geschieht aus dem „Expansion"-Schub. Diese Lösung ist meist billiger und erfüllt Ihren Zweck manchmal sogar besser, da die Treibersoftware dann gleich im Hauptspeicher steht und schneller auf sie zugegriffen werden kann. Ein Beispiel für den Einsatz eines solchen Treibers kann eine zusätzliche serielle Schnittstelle sein, die der Amiga grundsätzlich nicht verwalten kann. Erst über die entsprechende Software kann der neue Port dann angesprochen werden.

Die Installation ist denkbar einfach: Sie brauchen nichts weiter zu tun, als das entsprechende File in den „Expansion"-Schub zu legen, oftmals geschieht dies mit einem Installationsskript sogar automatisch (siehe: 5.1.4 Anwendungen). Beim nächsten Neustart werden dann alle Treiber automatisch eingebunden.

Für die Shell-Anwender sei hier noch eine kleine Ergänzung zu dem „automatisch" gegeben: Das Einbinden erfolgt mit dem Programm „binddrivers", das normalerweise in der „Startup-Sequence" aufgerufen wird. Wenn Sie diese Zeile jedoch entfernen, funktioniert in dieser Hinsicht gar nichts mehr. Außerdem sollten Sie selbst dann, wenn Sie die Workbench niemals benutzen, die „.info"-Files von Treibern immer mitkopieren. Denn in den meisten Fällen werden auch Daten aus diesen Files benötigt, die eben doch mehr als nur die Grafik für die Workbench enthalten.

5.1.3 Utilities

Während in den „WBStartup"-Ordner nur solche Utilities gehören, die im Hintergrund laufen, werden Sie sicherlich auch eine Reihe von kleinen Programmen besitzen, die nur kurzzeitig benötigt werden, um schnell irgendeine Aufgabe zu erfüllen. Ein Beispiel wäre ein Tool zum Anzeigen von Texten, wie etwa „More". Da neben den auf der Workbench bereits vorhandenen Utilities ja gerade im PD-Bereich eine reiche Auswahl unter solchen Hilfsmitteln besteht, ist eine geschickte Organisation gefordert, wenn Sie hier noch die Übersicht behalten wollen.

Das beginnt schon bei der „Installation". Utilities bestehen meist nur aus einem einzigen File, das Sie einfach in einen eignen Schub auf der Workbench schieben sollten. Bei der Namensgebung können Sie Ihrer Phantasie freien Lauf lassen; ob Sie den Schub nun „Utilities", „Tools", „Hilfsmittel" oder sonst irgendwie nennen, ist völlig egal, Hauptsache Sie wissen nachher, wo Sie das Programm wiederfinden können. Ab einer gewissen Anzahl von Tools sind auch Unterverzeichnisse zu empfehlen, um

Programme, die zu einem Themengebiet gehören, inhaltlich zu gliedern, z.B. Disketten-Utilities.
Neben dem Hauptprogramm kommen gelegentlich auch noch Libraries oder Devices hinzu. Grundsätzlich führt in einem solchen Fall kein Weg an den standardmäßig eingestellten Pfaden „Libs:" und „Devs:", etc., vorbei. Wie Sie diese zusätzlichen Files zu bestimmten Utilities von denen der Workbench dennoch trennen können, werden Sie im „Shell"-Teil erfahren. Den Workbench-Anwender interessiert so etwas meist wenig, da die Liste der Bibliotheken auf der Workbench normalerweise nicht erscheinen.
Schließlich gehören auch meist Dokumentationen zu den Programmen. Es ist hier ratsam, wieder einmal ein eigenes Verzeichnis anzulegen, am besten in dem Ordner, in dem Sie die Utilities selbst haben. Um reine Textfiles dann auch sofort anzeigen zu können, und nicht immer erst ein Anzeigeprogramm vorher starten zu müssen, sollten Sie den Namen Ihres bevorzugten Textviewers in dem „Default Tool"-Feld zu den Dokumentationen eingeben. Auch diese Informationen können Sie über die „Info"-Funktion aus dem „Icons"-Menü abrufen und editieren. Um beispielsweise den standardmäßigen „More" zu benützen, geben Sie hier „SYS:Utilities/More" an. Und schon genügt ein Doppelklick auf den Text selbst, um ihn auf den Bildschirm gebracht zu bekommen.
Utilities, die Sie ständig benötigen, sollten Sie mit der „Leave Out"-Funktion direkt in den Workbench-Screen verlagern, um sich die Arbeit zu ersparen, ein zusätzliches Window öffnen zu müssen. Allerdings sollte auch von dieser Möglichkeit nur maßvoll Gebrauch gemacht werden, damit Sie nicht schon auf der Workbench die Übersicht verlieren.

5.1.4 Anwendungen

Bei der Nutzung von Anwendungen muß zwischen Benutzern von Festplatten und solchen, die nur mit Disketten arbeiten, unterschieden werden. Während für einige Utilities meist noch genügend Platz auf der Workbench-Diskette ist (oder gemacht werden kann, siehe: 5.2.1 Platz schaffen), belegen Anwendungen meist eine ganze oder gar mehrere Disketten. Dann bleibt nicht viel zu tun, da Sie in diesem Fall entweder gleich von der entsprechenden Diskette starten oder nach dem Laden der Workbench die Disketten wechseln müssen und mit der Anwendungs-Diskette weiterarbeiten müssen.

Bei der Installation auf die Festplatte passiert da schon einiges mehr. Fast immer ist ein Installationsskript vorhanden, daß Ihnen das Kopieren der benötigten Files in die unterschiedlichen Verzeichnisse abnimmt. Da ist zunächst das programmspezifische Verzeichnis selbst, das erstellt und anschließen gefüllt wird, und dann werden noch Libraries, Devices, Treiber und alles mögliche auf der gesamten Platte verteilt. Dabei ist es meist sehr schwierig, noch den Überblick zu behalten, was nun eigentlich wohin kopiert wurde. Das aber müßte man wissen, wenn man die Anwendung später wieder von der Platte löscht, da sonst entweder nicht mehr benötigte Files übrig bleiben und kostbaren Platz verschwenden, oder Sie aus Versehen Files löschen, die gar nicht zu der entsprechenden Anwendung gehören. Die einzige Lösung dieses Problems ist es, auf der Originaldiskette nachzusehen, welche Files in Frage kommen.

Aber auch sonst ist das Installieren von Software nicht immer einfach. Meist unterscheiden sich die Programme/Skripts so stark voneinander, daß eine einheitlich

Bedienung nicht mehr möglich ist. Aus diesem Grund hat sich Commodore dazu entschlossen, nun doch endlich einmal ein Programm zu erstellen, das die Aufgabe zufriedenstellend löst und von allen Firmen verwendet werden kann, die eine Lizenz dafür erworben haben. Wenn Sie die Workbench 2.1 auf Festplatte installiert haben, sind Sie bereits ein erstes Mal damit in Berührung gekommen, da es bereits für diesen Zweck eingesetzt wurde.
Es zeichnet sich dadurch aus, daß zunächst eine von drei Anwenderebenen gewählt werden kann, je nach Ihrem Wissensstand. Abhängig von dieser Wahl verläuft die eigentliche Installation dann mehr oder weniger automatisch und gibt Ihnen entsprechende Möglichkeiten, einzugreifen. Dabei wird immer angezeigt, was als nächstes getan wird, wodurch schon einmal die Ungewißheit, was denn nun eigentlich vor sich geht, aus der Welt ist. Außerdem kann zu jeder Entscheidung ein Hilfstext abgerufen werden, der Ihnen die Konsequenzen möglicher Vorgehensweisen aufzeigt und dadurch Ihnen die Wahl erleichtert.

5.2 Optimierung der Workbench

5.2.1 Platz schaffen auf der Workbench

Der Wunde Punkt aller Disketten-Benutzer ist der Platz auf der Workbench-Disk. Sie wird bereits fast voll ausgeliefert und es sollen dennoch möglichst viele Utilities darauf passen. Und auch die Besitzer kleinerer Festplatten haben es nicht immer einfach, denn die Installation des kompletten AmigaOS 2.0 kostet schon ein paar Mega Bytes. Diese Situation verlangt förmlich nach einem Frühjahrsputz, um all die nicht benötigten Files los zu werden und sich frei entfalten zu können. Doch was kann man bedenkenlos löschen, was ist zum korrekten Arbeiten unerläßlich? Diese Fragen sollen geklärt

werden! Da im Shell noch einiges mehr gekürzt werden kann, folgen im Shell-Teil nochmals einige Tips, die allerdings nicht unbedingt so sehr für den Workbench-Anwender geeignet sind (siehe: 6.4.2 Platz schaffen im Shell). Und gerade der verlangt ja am meisten noch freiem Platz.
Da aber eines von Murphy's Gesetzen besagt, daß Sie ein File, das Sie soeben gelöscht haben, im nächsten Moment mit Sicherheit wieder brauchen werden, sollten Sie geeignete Vorkehrungen treffen. Dazu gehört selbstverständlich, daß Sie beim Löschen nur auf einer Sicherheitskopie arbeiten. Doch außerdem sollten Sie eine zweite, frisch formatierte Diskette zur Hand haben (nennen wir sie „Workbench_II"), auf die Sie alle Files, die Sie zu löschen beabsichtigen, kopieren können. Es wäre natürlich auch möglich, eine weitere Sicherheitskopie anzufertigen, doch die Absicht hinter der beschriebenen Vorgehensweise ist folgende: Wenn Sie nach und nach immer neue Software auf Ihre Workbench kopieren, die mit der Zeit vielleicht doch nicht mehr so dringend benötigt wird, so können Sie diese immer auf die zweite Diskette kopieren, ohne auch hier gleich wieder vor einem Platzproblem zu stehen. Wesentlich ist aber nur, daß beide Disketten zusammen alle Dateien der ursprünglichen Workbench enthalten, daß Sie zudem möglichst disjunkt sein sollten, ist darüber hinaus aber äußerst empfehlenswert.

Das erste Verzeichnis, das wir verstümmeln wollen, ist natürlich das „Prefs"-Directory. Kein anderer Ordner bietet sich dermaßen an, denn wenn Sie Ihre Preferences erst einmal optimal eingestellt haben, werden Sie wohl nur noch selten Änderungen vornehmen. Sie können also getrost alle Voreinsteller (nach dem Kopieren!) löschen, lediglich für den Fall, daß Sie keine Echtzeituhr besitzen, ist der „Time"-Einsteller unverzichtbar, denn die korrekte Zeiteinstellung wird von vielen Programmen vor-

ausgesetzt und erleichtert das Arbeiten im Filesystem auch ungemein. Dann sollten Sie „Time" aber auch mittels „Leave Out" auf die Workbench auslagern, um bei jedem Neustart sofort darauf zugreifen zu können.
Das nächste Opfer ist der „Utilities"-Schub. Utilities an sich sind ja nichts schlechtes, doch Sie werden im Laufe der Zeit sicherlich sinnvollere Programme aus dem PD-Bereich ansammeln, als z. B. „Clock", da erfüllt Ihre Digitaluhr bessere Dienste. Auch das Programm „Display" ist verzichtbar, wenn Sie nicht öfters irgendwelche IFF-Grafiken anzeigen wollen. „More" hingegen ist immer brauchbar, da eine der häufigsten Arbeiten am Computer im Lesen von Texten besteht. Wenn sich allerdings ein geeigneter Ersatz in Ihrem Besitz befindet, kann auch auf „More" verzichtet werden. Das „Tools"-Verzeichnis befindet sich unglücklicherweise gar nicht auf der Workbench-Diskette, weshalb man es auch nicht löschen kann. Wenn Sie jedoch das komplette Paket auf Ihre Festplatte installiert haben, finden Sie hier viele Programme, für die Sie gar keine Verwendung haben und die Sie folglich entfernen können. Andererseits könnte es auch sein, daß Sie als Disketten-Benutzer doch eines der „Tools" brauchen. Dann schaffen wir hoffentlich mit diesem Kapitel genügend Platz,so daß Sie dieses dann doch noch auf die Workbench quetschen können.

Auch im „System"-Ordner ist nicht allzuviel enthalten, das zu löschen es sich lohnen würde. Auf „NoFastMem" kann aber auf jeden Fall verzichtet werden. Auch „FixFonts" werden Sie wohl nur so selten benutzen, daß der Platzgewinn ein gelegentliches Diskettenwechseln rechtfertigt. Das Programm „Format" hingegen ist sicherlich unverzichtbar, zumindest solange Sie kein umfassenderes Disk-Utility besitzen, das auch das Formatieren einschließt. Bedenken Sie jedoch, daß das original „Format" sehr mächtig ist, da mit ihm nicht nur normale Disketten, sondern auch Festplatten und

alle möglichen anderen Speichermedien formatiert werden können.

Was noch bleibt, ist das „Devs"-Verzeichnis, das doch wieder hoffen läßt. Schließlich kann in vier Unterverzeichnissen doch einiges versteckt sein. Leider ist es aber vielmehr so, daß diese, wie z. B. das „Printers"-Directory darauf ausgelegt sind, daß Sie den gewünschten Druckertreiber hineinkopieren und nicht die überflüssigen löschen. Wie dem auch sei, Sie sehen sicher selbst, welche der enthaltenen Files Sie noch benötigen und welche nicht. Gerade bei den „DOSDrivers" gibt es da von Fall zu Fall starke Abweichungen, abhängig davon, was Sie mit Ihrem Amiga eigentlich anstellen.

5.2.2 Effizient arbeiten

Doch auch, wenn eine gewisse Bewegungsfreiheit Voraussetzung für effizientes Arbeiten ist, so genügt diese alleine doch nicht. Daß Geschwindigkeit aber auch keine Hexerei ist, ist altbekannt. Ein ganz wesentlicher Punkt um die Leistung zu steigern ist die Organisation Ihrer Dateien. Dabei ist Ihr Vorgehen wieder einmal davon abhängig, ob Sie eine Festplatte Ihr eigen nennen, oder nicht.
Als Disketten-Benutzer kommt es in erster Linie darauf an, die Zugriffszeit zu erhöhen. Dazu gehört zunächst, daß Sie das Angebot von AmigaOS 2.0 annehmen und unbedingt das FastFileSystem benutzen, da es eine enorme Geschwindigkeitssteigerung mit sich bringt (und zudem noch mehr Daten auf die Disk passen). Nur wenn die Diskette von anderen Amiga-Besitzern, die noch mit der alten Kick 1.3 arbeiten, gelesen werden soll, muß auf das OldFileSystem zurückgegriffen werden. Dies sollte aber den Ausnahmefall darstellen.

Ein weiterer Bremsfaktor fällt gerade bei nahezu vollen Disketten stark ins Gewicht: Einzelne Dateien werden über die gesamte Diskette verstreut und der Schreib-/Lesekopf muß ständig von Spur zu Spur fahren, um auf die Daten zugreifen zu können. Aber gerade dieser Vorgang ist besonders zeitintensiv. Vermeiden läßt sich das Problem jedoch nicht so einfach, da beim Löschen von alten Files zwangsläufig Lücken entstehen, die später aufgefüllt werden müssen. Abhilfe können jedoch Disketten-Optimierungsprogramme schaffen, die zusammengehörende Files auch nahe beieinander liegend anordnen. Wenn Sie kein solches Programm zur Verfügung haben, hilft folgender Trick oft auch schon weiter: Formatieren Sie eine Diskette und kopieren Sie sämtliche Files der alten Disk auf diese. Dies darf natürlich nicht mit dem Diskcopy-Befehl oder einem anderen Kopierprogramm geschehen, da dabei die alte Struktur erhalten bleibt.
Sie müssen vielmehr alle Dateien aus dem einen Disketten-Window in das andere ziehen. Dabei können dann nämlich alle Files in einem Stück geschrieben werden, da noch keine Lücken vorhanden sind, die es aufzufüllen gilt.
Auf einer Festplatte bringen derartige Tricks nur wenig, da die heutigen Platten meist ohnehin so schnell sind, daß die Daten innerhalb kürzester Zeit eingelesen sind. Hier macht sich dann eine logische Organisation der Daten eher bemerkbar. Dar der wesentliche Faktor der Mensch ist!
Es dauert eben seine Zeit, bis man sich durch die verschiedenen Unterverzeichnisse wühlt, egal wie schnell diese nun auf dem Bildschirm erscheinen. Deshalb ist es besonders wichtig, ein ausgewogenes Verhältnis zwischen der Anzahl der Verzeichnisse auf einer Ebene und der maximalen Tiefe zu finden. Denn entweder Sie kennen sich in einem Window vor lauter Schubladen nicht mehr aus, oder aber

müssen, um bis zu dem gewünschten File vorzustoßen, ein Dutzend Windows öffnen. Ein guter Kompromiß sind üblicherweise etwa fünf bis zehn Unterverzeichnisse in einem Verzeichnis, wobei natürlich ganz besonders auf das Hauptverzeichnis geachtet werden muß. Denn man neigt immer dazu, einen neuen Schub nach ganz oben zu legen, wenn man nichts findet, zu dem das neue Thema passen würde. Hier ist Disziplin und Ordnung gefordert.
Doch abgesehen von diesen allgemeinen Tips, die wohl auf jedem Rechnersystem gelten, haben Amiga-Besitzer ja den Vorteil eines genialen Betriebssystems. Und da gibt es dann eben solche Perlen wie die „Leave Out"-Funktion, mit der Sie die am häufigsten benötigten Programme direkt auf den Workbench-Screen legen können. Machen Sie auch von dieser Funktion sinnvollen Gebrauch, dann werden Sie im Normalfall überhaupt kein Window mehr öffnen müssen, was natürlich enorme Zeit spart, gleichgültig ob Sie nun mit Festplatte oder Disketten arbeiten. Natürlich ist auch hier wieder maßvolle Nutzung angesagt, da auch das Laden dieser Icons seine Zeit braucht, und wenn dann Programme erscheinen, die Sie nur selten benötigen, machen Sie Ihren Gewinn zunichte. Außerdem machen zu viele Icons im Workbench-Screen die Sache auch sehr unübersichtlich.

Und auch die zweite Möglichkeit, Programme sehr schnell zu starten, sollten Sie nutzen. Belegen Sie Ihre F-Tasten mit „FKey"! Denn dann werden Sie nicht einmal mehr zur Maus greifen müssen, um ein Programm zu starten. Auch das Laden der Icons fällt weg; effizienter geht es kaum.

Und noch einen letzten Tip sollen Sie mit auf den Weg bekommen: Während die genannten Empfehlungen sich immer auf das Starten von Programmen bezogen,

wozu die Workbench ja auch da ist, ist es natürlich auch interessant, was Sie mit diesen Programmen dann anstellen. Denn so schön Multitasking auch ist, ab fünf Screens oder zehn Windows wird es doch langsam unübersichtlich. Dann werden Utilities wie der „Autopoint" zum Fluch, da die Maus garantiert immer noch ein paar Pixel zur Seite rutscht, Sie das gewünschte Window verlassen und Ihre Texteingabe in das falsch Fenster geht. Hier ist dann nichts so Hilfreich wie das „Zoom"-Gadget, das gleich wieder für Ordnung sorgt. Dabei sollten Sie auch nicht zurückschrecken, die alternative Größe zu verändern. Ganz besonders über die Position läßt sich einiges bewerkstelligen. Wenn Sie beispielsweise mit mehreren Programmen arbeiten, aber immer nur ein Window bildschirmfüllend benötigen, dann sollten Sie die übrigen auf minimale Größe bringen und am unteren Rand des Screens nebeneinander aufreihen. Dann genügt Ihnen ein Klick auf das „Zoom"-Gadget des momentan aktiven Fensters, gefolgt von einem Klick auf das „Zoom"-Gadget des gewünschten und schon haben Sie das andere Programm wieder zur Verfügung. Kein lästiges Blättern durch eine Unzahl von Windows mehr.

Die Workbench sollte übrigens immer als Window dargestellt werden (d.h. „Backdrop" ausschalten), da es so einfacher ist, auf die Disketten-Icons zuzugreifen. Ansonsten müßten Sie immer alle anderen Windows zur Seite schieben, da ein Window niemals hinter einem Screen liegen kann.

Bei der intensiven Nutzung von Screens sollten Sie zudem eine F-Taste mit der Funktion des Screen-Blätterns belegen, vorausgesetzt Sie arbeiten häufiger mit der Tastatur als mit der Maus. Überhaupt sind Tastaturkürzel fast immer schneller als die Maus-Äquivalenten, die nur bei seltener Benutzung von Programmen durch ihr

intuitives Auftreten schneller zum Erfolg führen. Bei häufig wiederholten Vorgängen ist eine einzelne Taste schneller gedrückt und der Lernaufwand zahlt sich bald aus. Die Liste der Tips & Tricks ließe sich fast endlos fortführen, doch werden Sie sicher schon bald Ihren eigenen Stil entwickeln, um mit dem neuen, professionellen Amiga auch ebenso professionell arbeiten zu können.

6. Shell

Auch in der Shell, die sich ja aus dem ursprünglichen „Command Line Interface" (CLI) gemausert hat, hat sich einiges getan. Die kontinuierliche Verbesserung, die ja schon mit dem NewCon-Handler abzusehen war, hat einen neuen Höhepunkt erlangt. Command History, Line Editing und das direkte Ausführen von Script-Files sind nur einige Highlights. Außerdem wird jetzt der Inhalt einer Bildschirmseite zwischengespeichert, so daß alle Ausgaben wieder sichtbar werden, wenn man ein zwischenzeitlich verkleinertes Window wieder auf volle Größe bringt. Doch es geht noch weiter: Der gesamte Aufbau des Systems hat sich mit der neuen Kickstart so tiefgreifend verändert, daß auch so bekannte Dinge wie die „Startup-Sequence" ganz neue Aspekte enthält.

6.1 Das Shell-Window

Das Shell-Window, das erste was man beim Hochstarten zu Gesicht bekommt,

6.1.1 Textfunktionen

Nun können Sie in CLI-typischer Weise Befehle eingeben. Doch nachdem Sie eine Zeile getippt und mit Return bestätigt haben, können Sie diese jederzeit mit Cursor-Up wieder zum Vorschein bringen. Dabei wird zunächst die zuletzt eingegebene Zeile angezeigt, dann die davor usw. Doch damit nicht genug. Wenn es schon einige Befehle her ist, daß Sie etwas bestimmtes getippt haben, können Sie auch den Anfang der Zeile erneut eingeben und anschließend Shift-Cursor-Up drücken. Dann werden nur noch die Zeilen angezeigt, deren Anfang mit den eingegebenen Zeichen übereinstimmt. Darüber hinaus können Befehlszeilen zu jedem Zeitpunkt editiert werden, d.h. Sie können mit den Cursor-Tasten innerhalb der Zeile hin- und herfahren, Zeichen löschen oder neue einfügen. Außerdem können die Cursor-Left/-Right Keys ebenfalls mit der Shift-Taste kombiniert werden, was dann einen Sprung an den Anfang bzw. das Ende der Zeile bewirkt. Ein wenig kritischer wird es bei den folgenden Kombinationen:

Ctrl-W
löscht das Wort links vom Cursor

Ctrl-U
löscht alle Zeichen vor dem Cursor

Ctrl-K
löscht den Rest der Zeile hinter dem Cursor
und übernimmt diesen in eine Puffer

Ctrl-Y
fügt den mit Ctrl-K ausgeschnittenen Puffer ein

Ctrl-X
löscht die komplette Zeile

Darüber hinaus kann zum Kopieren von Texten auch die Maus verwendet werden. In wie weit dies in einer Tastaturorientierten Umgebung wie der Shell sinnvoll ist, müssen Sie selbst entscheiden. Da diese Funktion aber nicht nur in der Shell selbst, sondern in Console-Windows ganz allgemein anwendbar ist, können auch Daten zwischen unterschiedlichen Fenstern bewegt werden und das ist dann natürlich wieder eine gerissene Sache. Zu diesem Zweck halten Sie einfach die Shift-Taste gedrückt und kennzeichnen dann mit der linken Maustaste den zu markierenden Bereich. Dieser kann nun mit der Kombination rechte Amiga-Taste + C kopiert und mit rechte Amiga-Taste + V wieder eingefügt werden. Daran erkennen Sie ganz klar, daß der Amiga ein überirdischer Rechner ist: Kein menschliches Wesen würde sich jemals solche Tastaturkürzel einfallen lassen!

6.1.2 Window-Eigenschaften

Neben diesen Neuerungen im Shell-Window hat sich auch das Window selbst geändert, oder genauer gesagt, man kann unter AmigaOS 2.0 beim Öffnen von Shell-Windows noch eine Reihe von Optionen benutzen. Wenn Sie also beispielsweise mit „newshell" einen neuen Task erzeugen, lautet die vollständige Befehlszeile: „newshell

con:x/y/breite/höhe/titel/optionen from datei". Neu dabei ist der Parameter „Optionen". Mögliche Parameter dafür sind:

AUTO
wird das Window mit dem Close-Gadget geschlossen, so öffnet es sich automatisch wieder, sobald eine Eingabe oder Ausgabe stattfindet

CLOSE
das Window bekommt ein Close-Gadget (Alternative zu endshell)

BACKDROP
es wird ein Backdrop-Window erzeugt, d.h. es liegt hinter allen anderen Fenstern

NOBORDER
das Window wird völlig verstümmelt, vergessen Sie diesen Parameter!

NODRAG
das Window kann nicht gezogen werden

NOSIZE
die Zoom- und Size-Gadgets fehlen

SCREEN
das Window wird auf einem fremden Public-Screen geöffnet, dessen Name als weiteren Parameter angeben müssen

SIMPLE
beim Vergrößern des Windows werden vorher verdeckte Bereiche korrekt angezeigt (Grundeinstellung für Shell-Fenster)

SMART
das Gegenteil von SIMPLE: der Text fließt nicht nach

WAIT
das Window kann nur über das Close-Gadgets geschlossen werden.

WINDOW
es kann die Adresse eines bereits geöffneten Windows angegeben werden, welches dann als Shell verwendet wird; sollte nur von Programmierern angewendet werden (wenn überhaupt)

6.2 Neues im AmigaDOS

Doch nicht nur die äußere Erscheinungsform hat sich geändert, die dem geplagten Shell-Anwender doch zumindest das Leben schon sehr viel leichter macht. Auch der Umgang mit den Befehlen und selbstverständlich die Befehle selbst wurden verbessert. Gerade in der Kombination verschiedener Befehle, sei es mittels Pipes oder Skript-Dateien, liegt eine enorme Kraft. Durch geschickte Anwendung dieser Komponenten können viele Probleme gelöst werden, ohne erst spezielle Programme zu erstellen.

6.2.1 Die Pipe

Eine wesentliche Rolle spielen dabei „Pipes". Was aber sind nun eigentlich „Pipes" (dt.: Röhren)? Das Prinzip ist am besten an einem Beispiel zu erklären: Stellen Sie sich vor, Sie haben ein Verzeichnis, das nur Textfiles enthält und Sie möchten jetzt alle diese Texte auf den Bildschirm ausgeben. Dazu müßten Sie immer wieder „type filename" eingeben, wobei Sie „filename" gegen den Namen der einzelnen Files ersetzen müßten. Dabei wäre es doch viel einfacher, gleich die Ausgabe des einen Befehls als Eingabe an den anderen weiterzureichen, quasi durch ein Rohr vom einen zum anderen Befehl zu schicken. Nun, Sie ahnen es schon:
Genau das ist es, was man mit einer „Pipe" anstellt. Die Lösung des geschilderten Problems läuft einfach über den Aufruf „type 'dir' „. Beachten Sie dabei, daß das „dir" in sogenannte Backticks eingeschlossen ist, das ist das Zeichen, das auf der Tastatur links oben unterhalb der Esc-Taste zu finden ist. Dieses Symbol kennzeichnet nämlich die Röhre, und die Ausgaben des Programms, das zwischen den Backticks eingeschlossen steht, werden an das äußere Programm weitergeleitet. Genauer betrachtet: Das „dir" listet das Verzeichnis auf und liefert als Ausgabe die Filenamen, die dem „type" dann als Eingabe dienen. Natürlich hätten Sie das gleiche Ergebnis auch mit „type #?" erreichen können, aber nur, weil der „type"-Befehl Muster in den Filenamen verarbeiten kann. Es gibt andere Anwendungen, in denen die „Pipe" unverzichtbar ist.

Eine äußerst trickreiche Anwendung der Pipe besteht im Zusammenhang mit Skript-Files. Wie Sie im nächsten Kapitel noch sehen werden, ist es möglich, Skript-Files äußerlich ebenso ablaufen zu lassen, als seien es eigenständige Programme. Nur in einem Punkt unterscheiden sie sich, und das ist ihre Handhabung der Ein-/Ausgabe-

Umlenkung. Denn wenn Sie ein Skript, nennen wir es einmal „script", durch „script >output" aufrufen, werden keineswegs alle Ausgaben der Programme, die aus dem Skript gestartet werden, in das File „output" umgelenkt. Denn was hinter einem solchen Befehl ja eigentlich steckt, ist das „execute"-Kommando, („execute script >output"), und dessen Ausgabe wird umgelenkt - nur gibt „execute" im Normalfall nichts aus.

Nun nehmen wir aber einfach unsere Pipe zur Hand und gestalten unseren Aufruf ein wenig anders: „echo >output 'script' „. Nun werden alle Ausgaben von „script", egal ob sie vom „execute" oder von den gestarteten Programmen stammen, dem „echo" übergeben, und das lenkt diese dann in das „output"-File. Das klingt zwar vielleicht ein wenig um die Ecke gedacht, ist aber eine verblüffend einfache Methode, dieses nicht ganz selten auftretende Problem zu umgehen. Denn wenn Sie Anwendungen benutzen, die zwar den Aufruf externer Kommandos, nicht aber Skript-Dateien verarbeiten können, und die Ausgabe zur späteren Weiterverarbeitung in ein temporäres File umlenken, so können Sie diesen Trick heranziehen, um dennoch Skript-Files korrekt ausführen zu lassen.

Auch dazu nochmals ein Beispiel: Sie möchten in einer Anwendung, die die Möglichkeiten dazu bereitstellt, über ein externes Kommando ein Inhaltsverzeichnis ausgeben lassen. Dann würden Sie an entsprechender Stelle „dir" eingeben. Doch wenn Sie vielleicht vorher noch einen kleinen Text und das aktuelle Verzeichnis angezeigt bekommen möchten, ist ein Skript-File die ideale Lösung. Es müßte ja nur „echo Inhaltsverzeichnis" - „cd" und „dir" enthalten. Wenn Sie nun den Aufruf aus dem Programm wie oben gezeigt mit „echo 'script' „ vornehmen, kann das Anwendungsprogramm die Ausgabe problemlos umlenken und Sie brauchen nicht erst nach einem

Utility Ausschau halten das genau das tut, was Sie genauso gut in einem kurzen Skript bewerkstelligen können. Natürlich lassen sich auch noch komplexere Dinge machen, Hier sind Ihrer Phantasie keine Grenzen gesetzt. Und oftmals, wenn Sie irgend etwas nicht bewerkstelligen können, ist das Problem mit den „Pipes" ganz einfach zu lösen.

6.2.2 Skript-Dateien

Bereits im vorangegangenen Kapitel wurde ein wenig mit „Skript-Dateien" gearbeitet, aber man kann mit ihnen noch viel mehr machen. Es gibt zwei Gründe, warum sie so ausführlich behandelt werden: Weil sie eine wesentliche Veränderung erfahren haben und ein äußerst wesentlicher Aspekt, der schon in früheren AmigaDOS-Versionen existierte, plötzlich nicht mehr im Handbuch erwähnt wird.

Neu ist, daß Sie Skripts nun auch einfach über ihren Namen aufgerufen werden können und nicht erst der Umweg über das „execute"-Kommando nötig ist. Denn statt wie bisher „execute scriptfile" zu schreiben, genügt ein einfaches „scriptfile". Nun, nicht ganz. Sie müssen zudem das S-Flag (für Skripts) setzen, was mit dem „protect"-Kommando erfolgen kann: „protect scriptfile +s". Dabei bedeutet das „+", daß dieses Flag zusätzlich zu den bereits gesetzten hinzugefügt wird. Dies ist das Zeichen für AmigaDOS, daß es sich hier um kein ausführbares Programm handelt sondern ein interner Aufruf über „execute" erfolgen muß. Davon merkt der Anwender allerdings nichts mehr, da sich für ihn das Skript wie ein richtiges Programm verhält (bis auf eine Kleinigkeit, doch dazu, wie sich dieses Problem umgehen läßt siehe: 6.2.1 Pipes).
Zu einem „richtigen" Programm gehören natürlich auch Parameter. Wenn Sie in einer Skript-Datei mit Parametern arbeiten wollen, so muß die erste Zeile mit „.key"

(beachten Sie den Punkt am Anfang!) beginnen. Darauf folgen die Parameter, jeweils mit einem Komma voneinander getrennt. An jeden dieser Parameter kann optional noch ein „/a" oder „/s" angehängt werden, das wiederum durch ein zusätzliches „/k" ergänzt werden kann. „/a" bedeutet dabei, daß der Parameter unbedingt erforderlich ist, „/s" kennzeichnet ein Flag, das nur entweder gesetzt oder nicht gesetzt werden kann, abhängig davon, ob der Parameter angegeben ist oder nicht. Ist auch noch „/k" gesetzt, so muß der Name des Platzhalters vor dem Parameter stehen, wie man es beispielsweise von dem „as" im „join"-Kommando gewohnt ist. Die Parameter können dann im folgenden als Platzhalter eingesetzt werden, wobei der Name des Parameters dann in spitzen Klammern eingefügt sein muß. Einige Beispiele werden das verdeutlichen:

;ARCHIVE - setzt oder löscht das Archiv-Bit

```
.key filename/a,set/s

if <set>
protect <filename> +a
else
protect <filename> -a
endif
```

;MOVE-Kommando, Aufruf erfolgt mit: „move file1 to file2"

```
.key from/a,to/a/k
```

```
copy <from> to <to>
delete >nil: <from>
```

Sie sehen, das häufiger benötigte „Move" läßt sich in ein paar Zeilen selber machen. Dabei wird auf Korrektheit der Anzahl der Parameter geprüft und falls das „copy"-Kommando fehlschlägt, wird das ursprüngliche File nicht gelöscht, da die Skript-Datei im Falle eines Fehlers abbricht. Wenn Sie die ganze Sache noch ein wenig komfortabler gestalten möchten, können Sie dieses Abbrechen mit einem „failat 25" vor dem „copy" verhindern, müssen danach aber Ihre eigene Fehlerabfrage machen, damit in einem solchen Fall nicht das File gelöscht wird, ohne kopiert worden zu sein. Dies könnte so aussehen:

```
;besseres MOVE-Kommando

.key from/a,to/a/k

failat 25
copy >nil: <from> to <to>

if error
echo „Fehler aufgetreten!!!"
else
delete >nil: <from>
endif
```

Wie Sie sehen, können auch komplexere Strukturen entstehen und mit ein wenig mehr Aufwand ist es auch kein Problem, eine Vielzahl von Parametern korrekt zu verwalten. Zu dem Thema Parameter gibt es noch einiges mehr zu sagen, was auch noch getan wird. Allerdings kommen Sie in den meisten Fällen mit einem einfachen „.key" aus. Manchmal ist es aber auch notwendig, neben den Parametern, die übergeben werden können, Ersatzwerte parat zu haben, falls kein Parameter angegeben wurde. Dies ist natürlich auch möglich, wobei Ihnen sogar zwei Alternativen zur Verfügung stehen. Für die erste Variante weisen Sie nach der Definition der Parameterzeile mit „.key" mit dem Schlüsselwort „.def" einem Parameter seinen Default-Wert zu. Das Format lautet: „.def parameter default". Sie können aber auch erst bei der Verwendung des Parameters selbst den Default-Wert setzen, was den Vorteil hat, daß Sie an unterschiedlichen Stellen jedesmal einen anderen Ersatzwert angeben können. Zur Anwendung dieser Variante geben Sie zwischen den spitzen Klammern zusätzlich noch den Default-Wert an, und zwar abgetrennt durch ein Dollarzeichen. Auch hierzu wieder ein Beispiel:

```
;alternatives CD-Kommando

.key directory
.def directory ram:

cd <directory>


;noch eine Variante für CD, gleiche Funktion wie oben
```

.key directory

cd <directory$ram:>

Vielleicht sind Ihnen auch schon Bedenken gekommen, was die spitzen Klammern angeht. Denn da diese Zeichen ja auch zur Ein-/Ausgabe-Umlenkung herangezogen werden, könnte es zu Verwechslungen kommen. Und dies ist auch durchaus der Fall, nämlich dann, wenn der Name der Datei, in die umgelenkt werden soll, zusammen mit seinem Pfad eine gewisse Länge überschreitet. Falls dies einmal passieren sollte, können Sie jedoch die Sonderzeichen „<„ und „>" auch umdefinieren, und das geht mit den Befehlen „.bra" und „.ket". Nach diesen Schlüsselwörtern folgt dann das Zeichen, das von nun an die Parameter einleitet/beendet. Ein Beispiel:

```
.key directory/a
.bra {
.ket }

dir {directory}
```

Sie können selbstverständlich auch andere Zeichen als die geschweiften Klammern benutzen, die sich jedoch anbieten, da sie sonst praktisch nie Verwendung finden. Auch der „." als Einleitung der Parameterbefehle kann umdefiniert werden, was in der Praxis wohl kaum Anwendung findet. Deshalb sei das verwendete Schlüsselwort auch nur der Vollständigkeit halber erwähnt: „.dot". Nach einem „.dot !" müßten Sie beispielsweise „!key" statt „.key" schreiben. Und auch das „$"-Zeichen zur Angabe des

Default-Werts kann verändert werden, wozu Sie einfach „.dollar" und dann das gewünschte Symbol schreiben. Auch hierzu noch einmal ein kurzes Beispiel:

```
;eine neues FORMAT

.dot !
!key device,name
!dollar ß

format drive <deviceßdf0:> name <nameßEmpty>
```

Sie sehen also, mit Skripts kann man sich das Leben ganz gewaltig erleichtern. Gerade durch die Möglichkeit, Ersatzwerte zu verwenden, können viele Standard-Kommandos so erweitert werden, daß sie im Normalfall genau das tun, was für Sie das gebräuchlichste ist, Sie aber dennoch optional alle Parameter verändern können, sollte dies einmal nötig werden. Da es nicht mehr nötig ist, vor jedes Skript ein „execute" zu setzen, erfolgt der Aufruf genauso kurz wie bei Kommandos und Sie verlieren nicht wieder das, was Sie mit Skripts gewinnen: eine Ersparnis an Schreibarbeit.

6.2.3 Environment-Variablen

Auch nicht mehr ganz taufrisch, aber im Zusammenhang mit Skripts sehr wesentlich sind Environment-Variablen. Und sonderlich bekannt waren sie bis jetzt eigentlich nie. Einvironment-Variable sind Variablen, die von Programmen ebenso wie aus der Shell

heraus gesetzt oder abgefragt werden können. Dabei unterscheidet man zwischen lokalen und globalen Variablen. Lokale Variablen werden mit dem Kommando „set" gesetzt und mit „unset" gelöscht. Sie sind nur in dem Shell, in dem diese Definition gemacht wurde, sowie allen Shells, die wiederum aus diesem Shell gestartet wurden, bekannt. Globale Variablen werden mit „setenv" gesetzt und können von allen Prozessen aus angesprochen werden. Die Abfrage erfolgt über das „getenv"-Kommando und mit „unsetenv" können bereits definierte Variablen wieder gelöscht werden.

Eine Environment-Variable kann sowohl eine Zeichenkette wie auch einen Zahlenwert annehmen. Angesprochen wird sie dann entweder wie schon erwähnt mit „getenv" oder einfach, indem ein „$"-Zeichen unmittelbar vor den Namen gestellt wird. Zum besseren Verständnis all diesen Setzens, Abfragens und Löschens hier ein paar Beispiele:

```
1> setenv test „Die ist ein erster Test!"
1> getenv test
Dies ist ein erster Test!

1> setenv kbyte 1024
1> eval 75 * $kbyte
76800

1> setenv temp ABC
1> getenv temp
```

```
ABC
1> unsetenv temp
1> getenv temp
Objekt nicht gefunden

1> set lokal „Shell 1“
2> set lokal „Shell 2“   ;in einem anderen Shell (Task 2)
1> echo „$lokal“
Shell 1
2> echo „$lokal“
Shell 2
3> echo „$lokal“
$lokal                   ;in diesem Shell (Task 3) ist „lokal“ unbekannt!
```

Damit dürften die grundlegenden Funktionen klar sein. Neben diesen frei definierbaren Variablen existieren noch einige Systemvariablen, die äußerst hilfreich sein können. Im einzelnen sind dies:

ECHO

wenn Sie „set echo on“ definieren, werden beim Abarbeiten von Skript-Dateien die einzelnen Befehle vor ihrer Ausführung angezeigt

PROCESS
gibt die Prozessnummer des aktuellen Shell-Windows an

RC
enthält den Rückgabewert des zuletzt ausgeführten Befehls; diese Information kann sonst nur mit „if" abgefragt werden

RESULT2
gibt die Fehlernummer des letzten Befehls an; kann mit „fault" in Klartext übersetzt werden

Mit diesen Variablen kann eine Menge angestellt werden, exemplarisch sei hier die korrekte Abfrage von „RESULT2" angegeben, wie sie sich vorzüglich für Skript-Dateien eignet:

fault 'echo „$reslut2" '

Dabei ist das „echo"-Kommando wieder einmal in Backticks eingeschlossen, da eine Pipe verwendet wird (siehe hierzu: 6.2.1 Die Pipe).

6.3 Kommandos

Da sprechen wir nun die ganze Zeit von der Kombination von Kommandos und Kommandofolgen, doch von den Kommandos selbst wurde noch keines richtig

beschrieben. Dabei wurden auch gerade die Kommandos, in jeder Hinsicht, stark verbessert. Die Leistungssteigerung beruht dabei auf drei Pfeilern: einer einheitlichen Benutzerschnittstelle, der Verlegung ständig benötigter Befehle in die Kickstart und die Erweiterung des vorhandenen Befehlssatzes. Die Beschreibung des für einen Blitzeinstieg bei weitem zu umfangreichen Befehlssatzes muß leider ausbleiben, doch zum Nachschlagen einzelner Funktionen eignet sich das Systemhandbuch ohnehin besser. Vielmehr sollen Sie im Folgenden den prinzipiellen Umgang mit den Befehlen erlernen, dann werden Sie sich bei der Handhabung mit allen Kommandos leichter tun.

6.3.1 Parameterschablone

Die Benutzerschnittstelle, die sich bei Kommandos in der Parameterschablone äußert, ist jetzt durchweg einheitlich. Sie bekommen von jedem Kommando durch den Aufruf „Programmname ?" eine genaue Definition der möglichen Parameter geliefert. Diese beschreibt, welche Daten unbedingt angegeben werden müssen, welche optional sind und in welchem Format sie übergeben werden. So bekommen Sie nach Eingaben von „dir ?" als Antwort „DIR,OPT/K,ALL/S,DIRS/S,FILES/S,INTER/S:" geliefert. Ähnliche Angaben wurden auch unter früheren Betriebssystemversionen geliefert, jedoch nicht von allen Befehlen. Außerdem wurde das Format beim AmigaDOS 2.0 nochmals erweitert und stärker standardisiert.

Es werden jeweils die Schlüsselwörter für die einzelnen Parameter aufgeführt, möglicherweise gefolgt von einem oder mehreren Schrägstrichen, hinter denen dann immer ein Buchstabe steht, der Auskunft über den Typ des Parameters gibt. Dabei stehen zur Auswahl:

/S
bei dem Schlüsselwort handelt es sich um einen Schalter; es kann entweder angegeben werden oder eben nicht, es folgen keine zusätzlichen Parameter

/K
das Schlüsselwort (Keyword) muß vor dem eigentlichen Parameter stehen, der durch ein „=" oder Leerzeichen davon getrennt wird

/N
es sind nur Zahlenwerte (Numbers) als Parameter gültig

/T
das Schlüsselwort arbeitet als Umschalter (Toggle) und invertiert einen voreingestellten Wert; funktioniert wie ein Schalter (/S)

/A
der Parameter an dieser Stelle muß unbedingt angegeben werden

/F
der gesamte Rest der Zeile wird als Parameter für dieses Feld übernommen, selbst wenn weitere Schlüsselwörter folgen sollten

/M
es können an dieser Stelle mehrere (Multiple) Parameter angegeben werden; dabei

werden all die Parameter diesem Feld zugeordnet, die zu keinem anderen passen; insbesondere unbedingt benötigte Felder (/A) werden zuerst aufgefüllt

Die meisten dieser Flags können miteinander kombiniert werden. Sollte hinter einem Schlüsselwort einmal überhaupt keine weitere Spezifizierung stehen, so bedeutet dies einfach, daß Sie an dieser Stelle entweder einen Parameter angeben können oder nicht, wie es etwa beim ersten Parameter des „dir"-Kommandos der Fall ist.
Um besser verstehen zu können, was bestimmte Kombinationen von Flags bedeuten, die zudem noch mit weiteren Parametern kombiniert sind, sollen wieder einige Beispiele mit bereits bekannten Kommandos folgen:

addbuffers ?
DRIVE/A,BUFFERS/N:

Bei „addbuffers" muß (/A) ein Device angegeben werden und es kann, muß aber nicht, als weiterer Parameter eine Zahlenwert (/N) folgen, der die Anzahl der Buffers bestimmt.

filenote ?
FILE/A,COMMENT,ALL/S,QUIET/S:

Auch „filenote" verlangt auf jeden Fall einen Parameter, diesmal einen Filenamen (/A), Sie können anschließend ein Kommentar angeben, können dies aber auch weglassen ( ), wodurch das alte gelöscht wird. Bei „all" und „quiet" handelt es sich um Schalter (/

S), die durch Angabe des jeweils entsprechenden Schlüsselworts gesetzt werden.

```
copy ?
FROM/M,TO/A,ALL/S,QUIET/S,BUF=BUFFER/K/N,CLONE/S,DATES/S,NOPRO/
S,COM/S,
NOREQ/S:
```

Bei „copy" wird es langsam unübersichtlich, doch die komplexe Parameterzeile läßt sich durch schrittweises Vorgehen nach und nach analysieren. So kann man z. B. alle Parameter angeben oder auch keinen. Das mag etwas verwundern, denn eigentlich ist man ja gewöhnt, eine Quelle und ein Ziel beim Kopieren anzugeben. Doch es geht auch anders. So kopieren Sie beispielsweise mit „copy to ram:" alle Dateien aus dem aktuellen Verzeichnis in die RAM-Disk. In diesem Fall ist es allerdings unerläßlich, das Schlüsselwort „to" anzugeben, da sonst „ram:" als „from" aufgefaßt wird und das „all"-Feld nicht gefüllt werden kann. Dieses muß aber in jedem Fall angegeben werden (/A). Wenn Sie mindestens eine Quell-Datei angegeben haben, so können Sie auf das „to" verzichten, denn dann ist der letzte Parameter, der keinem der anderen noch ausstehenden Felder zugeordnet werden kann, das Zielverzeichnis. So ist der Aufruf „copy c:dir c:list ram:" durchaus zulässig. Bei allen noch folgenden Parametern, die durch ein „/S" gekennzeichnet sind, handelt es sich wieder um Schalter, die keiner weiteren Erläuterung mehr bedürfen. Was aber noch interessant ist, ist das Feld „BUF=BUFFER/K/N". Nicht nur, daß hier zwei alternative Schlüsselwörter angegeben sind („buf 10" ist gleichbedeutend mit „buffer 10"), auch finden wir hier zum ersten Mal die Kombination von Flags vor. „/N" kennen wir schon, es gibt an, daß der Parameter eine Zahl sein muß. Um die Verwechslung mit einem Filenamen auszuschließen (das

z.B. „10" heißen könnte), fordert dieses Kommando mit dem „/K" das Voranstellen des Schlüsselworts vor dem Parameter. Dieses Vorgehen werden Sie bei allen Kommandos vorfinden, die mit „multiple"-Parametern (/N) arbeiten. Denn während normalerweise von einer gewissen Ordnung ausgegangen werden kann, weiß man hier ja vorher nie, wie viele der Parameter nun zu diesem Feld gehören. So bedeutet „copy a b c 10 d", daß die Files „a", „b", „c" und das File „10" in das Verzeichnis „d" kopiert werden sollen. Erst durch die korrekte Angabe „copy a b c buf 10 d" wird klar, daß Sie einen Buffer wählen wollen und nicht ein File namens „10" ansprechen. Dadurch wird es auch wieder möglich, den „buffer"-Parameter vor dem „to"-Parameter anzugeben. Durch diese Vorgehensweise sind alle möglichen Parameter mit Ausnahme von „from" und „to" eindeutig einzuordnen und die restlichen werden einfach so aufgeteilt, daß „to" den letzten Parameter bekommt und „from" alle übrigen.

join ?
FILE/M/A,AS=TO/K/A

An dem kompakten Beispiel des „join"-Befehls wird eine andere Vorgehensweise klar. Hier werden unbedingt (/A) ein oder mehrere (/M) Quellfiles verlangt, und außerdem zwingend (/A) ein Zielfile, dem eines der beiden Schlüsselwörter „as" oder „to" vorangestellt sein muß (/K). Diese Einschränkung ist eigentlich unnötig, da AmigaDOS inzwischen schlau genug ist, selbst herauszufinden was Sie wahrscheinlich wollen. Aber wie das mit fehlertoleranten Systemen so ist: Sie tolerieren eben leider auch Fehler, die wirklich solche sind. Deshalb wird an dieser Stelle ein Schlüsselwort gefordert, denn nur zu leicht vergißt man, ein Zielfile anzugeben und dann würde das letzte der Quellfiles überschrieben werden. Also eine weitere Sicherheitsmaßnahme,

die uns hier aber auch vorzüglich als Beispiel diente.

6.3.2 Residente Kommandos

Viele Kommandos gehören einfach zum Standardbefehlssatz und gerade im Zusammenhang mit Skript-Files, in denen gewisse Konstruktionen ständig vorkommen, beschleunigt deren Präsenz im Speicher die Abarbeitung enorm. Gerade Diskettenbenutzer werden daher überaus dankbar sein, daß viele der am häufigsten benötigten Kommandos in die Kickstart verlegt wurden. Sie können natürlich nach wie vor mit dem Befehl „resident" zusätzliche Kommandos im Speicher halten, aber diese müssen zumindest einmal geladen werden, was die bereits vorhandenen residenten Kommandos nicht mehr müssen. Eine vollständige Liste der momentan residenten Kommandos erhalten Sie durch Aufruf von „resident" ohne Parameter. Dabei wird auch angegeben, ob es sich um ein internes Kommando handelt („INTERN"). Hier ein Überblick:

Alias
Set
Run

Ask
Get
Setenv
CD

Getenv
Skip

Echo
If
Stack

Else
Lab
Unalias

EndCLI
NewCLI
Unset

EndIf
NewShell
Unsetenv

EndShell
Path
Why

EndSkip
Prompt

.ket

Failat
Quit
.bra

Fault
Resident
.key

6.3.3 Der Ed

Um für den nächsten Abschnitt, die Modifikation der „Startup-Sequence" gerüstet zu sein, wird ein kleiner Abstecher zum „Ed", einem der beiden mitgelieferten Editoren, nötig. Er eignet sich vorzüglich zum Editieren kurzer Text-Dateien, wie etwa Skript-Files, insbesondere der „Startup-Sequence". Da auch der „Ed" einige Verbesserungen erfahren hat, zu denen unter anderem die Einführung von Pull-Down-Menüs gehört, hat sich die Bedienung stark vereinfacht. Sie müssen dem Editor zwar immer noch als Parameter den Namen des Files, das Sie editieren möchten, angeben, obwohl es oftmals viel angenehmer wäre, diesen erst nach dem Programmstart komfortabel über einen Filerequester anzugeben. Doch sonst stehen Ihnen die wesentlichen Funktionen eines Texteditors über die Menüs zur Verfügung. Selbstverständlich sind auch noch die bisherigen Tastaturkürzel, wie etwa Esc-X zum Verlassen des Programms mit Abspeichern, zur Verfügung. Und falls Sie intensiver

mit diesem Editor arbeiten wollen, werden Sie wohl auch nicht an den Tastaturfunktionen vorbeikommen, da nur auf diese Weise flüssiges Arbeiten möglich ist. Doch für unsere Zwecke reichen die Menüs völlig aus.

Im „Project"-Menü finden Sie alle File-Operationen. Hiervon benötigen Sie im wesentlichen nur „Save" zum abspeichern bzw. „Save & Exit" zum Abspeichern und anschließenden Verlassen des Editors. Sehr hilfreich sind überdies die Funktionen des „Edit"-Menüs. Hier sind Blockoperationen möglich, die es Ihnen erlauben, Textteile zu kopieren, verschieben oder zu löschen. Da oftmals auch nur einzelne Zeilen entfernt werden müssen, existiert auch hierzu ein Menüpunkt, der aber praktischerweise von der Tastatur aus aufgerufen werden sollte. Sie finden die entsprechenden Äquivalenten ja ohnehin am Ende der Zeile jedes Menüpunkts angegeben, wobei „Esc" ein vorheriges Betätigen der „Esc"-Taste verlangt, woraufhin zur Eingabe des eigentlichen Befehls am unteren Ende des Windows ein Sternchen („*") erscheint und das Dach („^") auf gleichzeitiges Drücken der „Ctrl"-Taste verweist. Somit müssen Sie zum Löschen einer einzelnen Zeile lediglich Ctrl-B drücken.

Die Menüs „Movement" und „Search" sind für unsere Anwendung wenig interessant, da Skript-Dateien üblicherweise nicht so lang sind, daß man sie nicht mehr überblicken könnte. Und auch die „Settings" und „Commands" werden erst bei intensiverem Gebrauch des „Ed" sinnvoll einsetzbar sein, doch sind ihre Funktionen so selbsterklärend, daß auch keine weiter Hilfestellung vonnöten ist.

6.4 Anwendung der Shell

So schön die Workbench auch sein mag, es gibt immer wieder Gelegenheiten, bei denen der Zugriff auf die Shell entweder unvermeidbar oder doch zumindest sehr viel effektiver ist. Gerade in Kombination mit Skript-Files kann diese Effizienz noch weiter gesteigert werden, so daß es durchaus verständlich ist, wenn auch heute noch viele Anwender die Shell der Workbench vorziehen. Ob Sie auch zu diesem Kreis gehören oder nicht, spielt allerdings hier keine Rolle, denn Sie sollen ja nur das nötigste Wissen vermittelt bekommen, das jeder Anwender früher oder später braucht.

6.4.1 Optimierung der Startup-Sequence

Der häufigste Grund in die Shell zu wechseln ist das Editieren der „Startup-Sequence". Nicht, daß man diese nicht auch von der Workbench aus editieren könnte, doch sie enthält ja ausschließlich Shell-Kommandos bzw. startet Programme aus der Shell heraus. Und genau um die geht es jetzt, denn der erste Ansatzpunkt zur Optimierung ist genau hier gegeben. Aus Gründen der Allgemeinheit enthält die Standard-"Startup-Sequence" nämlich viel überflüssigen Ballast, den Sie vielleicht gar nicht nutzen können. Deshalb wollen wir uns nun eben diese vornehmen und sehen, was man an ihr verbessern könnte.

Um auf einer gemeinsamen Basis zu stehen, folgt nun eine Auflistung der „Startup-Sequence" der Workbench 2.0, die zum besseren Verweis mit Zeilennummern versehen wurde. Welche Änderungen bis zur endgültigen „Startup-Sequence" der Workbench 2.1 noch vorgenommen werden, kann man noch nicht absehen, doch

werden auch diese mit Sicherheit nicht allzu tiefgreifend sein. Selbstverständlich werden bei dieser Gelegenheit auch wieder ein paar neue Kommandos der Workbench 2.0 erläutert.

```
01: c:setpatch >NIL:
02: c:version >NIL:
03: addbuffers >NIL: df0: 15
04: Failat 21
05:
06: resident >NIL: c:List pure add
07: resident >NIL: c:Copy pure add
08: resident >NIL: c:Assign pure add
09: resident >NIL: c:Execute pure add
10:
11: makedir ram:T ram:Clipboards ram:env ram:env/sys
12: copy >NIL: ENVARC: ram:env all quiet noreq
13:
14: assign ENV: ram:env
15: assign T: ram:t ;set up T: directory for scripts
16: assign CLIPS: ram:clipboards
17: assign REXX: s:
18:
19: if exists sys:Monitors
20:     join >NIL: sys:monitors/~(#?.info) as t:mon-start
```

```
21: execute t:mon-start
22: delete >NIL: t:mon-start
23: endif
24:
25: BindDrivers
26:
27: setenv Workbench $Workbench
28: setenv Kickstart $Kickstart
29:
30: IPrefs
31: ;
32: echo „Amiga Release 2.  Kickstart $Kickstart, Workbench $Workbench“
33:
34: conclip
35:
36: mount speak:
37: mount aux:
38: mount pipe:
39:
40: path ram: c: sys:utilities sys:rexxc sys:system s: sys:prefs
    sys:wbstartup add
41: if exists sys:tools
42: path sys:tools add
43: if exists sys:tools/commodities
```

```
44: path sys:tools/commodities add
45: endif
46: endif
47:
48: ;If this is the initial boot (i.e. keyboard env variable is not set)
49: ;then execute PickMap which will query for a keymap and then set the
50: ;keyboard env variable.
51: ;
52: ;if keyboard env variable is set, set the keymap
53: if ${sys/keyboard} NOT EQ „*${sys/keyboard}“
54: setmap ${sys/keyboard}
55: else
56: PickMap.hd sys:
57: endif
58:
59: if exists s:user-startup
60: execute s:user-startup
61: endif
62:
63: rexxmast >NIL:
64:
65: LoadWB
66:
67: endcli >NIL:
```

Wie Sie sehen, ein recht langes Skript-File. Doch Sie werden bei weitem nicht alles benötigen. Aber beginnen wir mit dem Anfang, der Zeile 1:

Das „setpatch"-Kommando hat seit jeher die Funktion, kleinere Fehler der Kickstart zu beheben, die zum Zeitpunkt deren Fertigstellung noch nicht bekannt waren, wohl aber vor der Herausgabe der Workbench. Deshalb sollte dieser Befehl immer zu Beginn der „Startup-Sequence" ausgeführt werden. Beim Erscheinen einer neueren Workbench braucht „setpatch" im C: Verzeichnis dann lediglich durch die aktuelle Version ersetzt zu werden. Daß die Ausgabe ins NIL: (also ins Nichts) umgelenkt wird, ist zum einen eine optische Angelegenheit, da es ja eigentlich nicht nötig ist, bei jedem Neustart immer wieder mitgeteilt zu bekommen, was „setpatch" gemacht hat. Doch im Zusammenhang mit der Workbench 2.0 und den neuen Preferences hat diese Variante noch einen gewichtigeren Grund: Wenn das „IPrefs"-Kommando ausgeführt wird, nachdem ein Bildschirm eröffnet wurde, so kann es seine Aufgabe nicht korrekt erledigen. Da der Aufruf dieses Programms aber erst in Zeile 30 erfolgt, darf keine vorherige Textausgabe stattfinden, da diese das Öffnen eines Bildschirms erzwingen würde. Bedenken Sie dies, wenn Sie die „Startup-Sequence" nach Ihren Wünschen verändern.

Auch die zweite Zeile offenbart einen neuen Kniff. Denn was auf den ersten Blick völlig unsinnig erscheint, hat in Wirklichkeit einen sehr sinnvollen Nebeneffekt. Denn eigentlich gibt das „version"-Kommando lediglich die aktuelle Kickstart- und Workbench-Versionsnummern aus. Wenn diese ins Nichts gelenkt werden wird der Befehl natürlich unsinnig. Doch nebenbei passiert noch etwas unbemerktes: Die lokalen Variablen „Kickstart" und „Workbench" erhalten die entsprechenden Versionsnummern.

Und nun ein kleiner Sprung: In den Zeilen 27 und 28 werden aus diesen lokalen Variablen globale Variablen gemacht, indem der Wert der lokalen Variablen mit dem „$"-Zeichen abgefragt und dem „setenv"-Kommando den entsprechenden globalen Variablen zugewiesen werden. Außerdem werden diese nun globalen Informationen in der Zeile 32 nochmals herangezogen, um die Versionsnummern in einem verständlichen Kontext auszugeben. Wenn Sie nicht gerade zwischen unterschiedlichen Kickstart-/Workbench-Versionen umschalten wollen, können Sie problemlos auf all die Befehlszeilen verzichten, die die Versionsnummern betreffen (2,27,28,32).
Das „addbuffers"-Kommando in Zeile 3 bewirkt, daß für das interne Laufwert 7,5 KByte Pufferspeicher reserviert werden (je Buffer 512 Byte). Wenn Sie noch ein zweites Laufwerk besitzen, sollten Sie einen entsprechenden Befehl für dieses einfügen.

Bei den bisherigen Kommandos konnten im Normalfall keine Fehler auftreten, um aber im folgenden sicher zu gehen, folgt in Zeile 4 ein „Failat 21", der den Abbruch des Skripts beim Auftreten von Fehlern unterbindet. Auch die maximal gültige Fehlernummer von 20 wird abgefangen.

In den Zeilen 6 bis 9 werden anschließend vier häufiger verwendete Befehle resident gemacht, was in erster Linie einer schnellere Abarbeitung der „Startup-Sequence" dient. Bei Festplattenbesitzern ist der Gewinn meist sehr gering, doch kosten die paar Befehle auch nicht allzuviel Speicherplatz, so daß man sie getrost behalten kann.
Die Zeilen 11 und 12 schaffen die nötige Umgebung für das System. Dazu gehört das Erstellen von vier temporären Verzeichnissen in der RAM-Disk (die dabei automatisch eingerichtet wird). Diesen werden dann in den Zeilen 14 bis 16 logische Verzeichnisse

zugewiesen. Dazwischen kopiert Zeile 12 das ENVARC:-Directory in den eben geschaffenen ENV-Ordner. Hierzu einige Erläuterungen: T: wird (wie das Kommentar in Zeile 15 auch aussagt) bevorzugt von Skript-Dateien benutzt. Die Verlegung ins RAM bewirkt dann natürlich eine schnellere Abarbeitung. CLIPS: ist ein neues Verzeichnis, in das beispielsweise die Texte gehen, die Sie aus der Shell ausschneiden. ENVARC: und ENV: beherbergen die Settings Ihrer Preferences. ENVARC: besitzt normalerwiese den Pfad SYS:prefs/env-archive. Dort werden die Einstellungen abgelegt, wenn Sie sie abspeichern.
Wählen Sie jedoch „Use", so werden sie ebenfalls abgespeichert, aber in das ENV:-Verzeichnis, das ja, wie gesehen, im RAM: liegt. Deshalb gelten diese Einstellungen nur temporär. Der Lese-Zugriff erfolgt immer auf das ENV:-Verzeichnis, weshalb die Daten aus ENVARC: beim Systemstart auch dorthin kopiert werden müssen, was genau das ist, was in Zeile 12 passiert. Schließlich wird auch noch REXX: auf S: gelegt, was ja auch sinnvoll ist, da ARexx Programme auch nichts anderes als eine Sonderform der Skript-Dateien, die ja im S:-Verzeichnis aufgehoben sind.

Der nächste Abschnitt (Zeilen 19 bis 23) ist weniger von seiner Funktion her als von der Realisierung interessant, denn er tritt bereits bei der Workbench 2.1 nicht mehr auf. Doch es ist sehr lehrreich, zu verstehen, wie hier vorgegangen wird. Da werden zuerst alle Files aus dem Verzeichnis sys:monitors mit Ausnahme der „.info"-Files zu einem neuen File im T-Verzeichnis verkettet. Das dazu nötige Namensmuster lautet: ~(#?.info), wobei die Tilde (~) für das „außer" steht und in der Klammer alle „.info"-Files selektiert werden. Das neue File wird anschließend als Skript-File ausgeführt, was schon vermuten läßt, daß auch die ursprünglichen Files alle Skipt-Dateien waren. Bei der Workbench 2.1 ist dies nicht mehr so, weshalb auch nicht näher auf den Sinn des

ganzen eingegangen werden soll. Doch es ist sicherlich eine gute Vorgehensweise wenn es darum geht, mehrere Skript-Files ausführen zu lassen und man von ihnen nur weiß, daß sie alle in einem Verzeichnis stehen, nicht aber, wie sie heißen. Solange Sie allerdings keine Monitor-Files im Monitors-Ordner haben, können Sie auf diese Sequenz verzichten, oder aber Sie können die Namen der vorhandenen Files gleich von Hand angeben. Dies ist allerdings nur Diskettenbenutzern anzuraten, da hier jeder überflüssige Befehl Zeit kostet. Die Festplattenbesitzer sollten mehr Wert auf eine offene Struktur legen. Denn wenn Sie an dieser Stelle entsprechende Änderungen vornehmen, ist es nicht mehr ausreichend, einfach ein neues Monitor-Icon in den Monitors-Schub zu ziehen, sondern Sie müssen wieder die „Startup-Sequence" abändern.

Das nächste Kommando in Zeile 25, „binddrivers" bindet alle im Expansion-Verzeichnis vorhandenen Gerätetreiber ein. Solange Sie keinen solchen Treiber besitzen können Sie sich diesen Befehl sparen, doch wehe Sie entfernen Ihn und denken bei der Installation neuer Treiber nicht mehr daran, ihn wieder einzufügen - dann haben Sie es mit einem nur schwer auffindbaren Fehler zu tun!

Nachdem die Bedeutung der beiden „setenv"- und des „echo"-Kommandos bereits erklärt wurde und auch die Funktion von „IPrefs" als Preferences-Steuerprogramm inzwischen hinreichend klar sein sollte, können wir uns gleich dem nächsten neuen Befehl, „conclip" in Zeile 34 zuwenden. Er wird zur Nutzung der Clipboard-Funktionen benötigt und sollte nicht fehlen. Da er auch die iffparse.library und das clipboards.device benötigt, sollten Sie sicherstellen, daß sich diese beiden im LIBS:- bzw. DEVS:-Verzeichnis befinden.

In den Zeilen 36 bis 38 werden Speak:, Aux: und Pipe: gemountet. Sie belegen alle nicht sehr viel Platz, können also bedenkenlos in der „Startup-Sequence" verweilen, doch werden Aux: und Speak: nur in den seltensten Fällen benötigt. Aux: ist das Device, das angesprochen wird, um ein Shell auf dem seriellen Port zu errichten - etwas, was wohl nicht einmal ein Prozent der Amiga-Besitzer benötigt. Und Speakl: ist nur dann brauchbar, wenn Sie unbedingt die grauenhafte Sprachausgabe genießen möchten, die einige Programme offerieren. Da dies aber ohnehin selten ist, können Sie, gerade wenn Sie von Diskette starten, wertvolle Zeit sparen und auch diese Zeile löschen.

In den Zeilen 40 bis 46 werden verschiedene Verzeichnisse dem Suchpfad hinzugefügt. Standardmäßig wird ja nur im C:-Directory nach Befehlen Ausschau gehalten und wenn Sie nun eines der Programme starten wollen, die im System-Ordner liegen, müssen Sie entweder den vollen Pfad angeben bzw. in diesem Verzeichnis stehen oder das Verzeichnis zum Suchpfad hinzufügen. Bevor allerdings das Tools-Directory und in diesem das Commodities-Directory hinzugefügt werden, erfolgt noch eine Überprüfung, ob diese überhaupt existieren. Denn das Tools-Verzeichnis wird zwar bei der Festplatteninstallation ins Haupt-Directory kopiert, doch auf der Workbench-Diskette selbst befindet es sich nicht. Deshalb können Sie ggf. auch überflüssige Zeilen entfernen, wobei immer darauf geachtet werden muß, eine komplette Klammer-Ebene zu entfernen, d.h. zu dem „if" auch das korrespondierende „endif" und der komplette Teil dazwischen.

Es gibt unter AmigaDOS 2.0 noch eine andere Variante, dem System mitzuteilen, wo sich noch weitere Programme befinden könnten: Mit dem assign-Kommando lassen

sich von nun an zu einem logischen Verzeichnis nicht nur ein, sondern mehrere reale Verzeichnisse zuordnen. Und das gilt natürlich auch für das C:-Verzeichnis. So könnte man beispielsweise „assign c: sys:system add" schreiben. Der Unterschied ist der, daß bei Verwendung des „assign"-Kommandos der zusätzliche Pfad auf AmigaDOS-Ebene mit dem C:-Directory in Bezug gesetzt wird. Das „path"-Kommando wirkt schwächer, es bezieht sich nur auf das jeweilige Shell und alle von ihm gestarteten Prozesse, was im Normalfall auch genügt.

In den Zeilen 48 bis 57 sehen Sie wieder einmal eine Anwendung von Umgebungsvariablen, diesmal zum Entscheiden, ob nach der Tastaturbelegung gefragt werden soll oder nicht. Die Vorgehensweise sollte inzwischen klar sein, außerdem ist dieser Teil des Skripts darüber hinaus noch dokumentiert (wobei Sie die Dokumentation natürlich zum Zwecke der Übersichtlichkeit entfernen können).

Weiter geht es in den Zeilen 59 bis 61. Dieser Teil bewirkt, daß nach dem File „s:user-startup" gesucht wird, und sollte es existieren, dies auch abgearbeitet wird. Dahinter verbirgt sich eigentlich die Absicht, den Anwender so weit als möglich aus der „Startup-Sequence" rauszuhalten, der mit diesem Kapitel natürlich entgegengearbeitet wird. Denn wenngleich es sicher sinnvoll ist, in einem System mit schneller Festplatte und noch schnellerem Prozessor möglichst viele Eventualitäten von vornherein einzuplanen und später nichts mehr an Systemdateien zu ändern, so liegt die Situation bei einem A500+ mit einem Diskettenlaufwerk doch etwas anders. Hier ist man darauf angwiesen derart umständliche Dinge wie die „Startup-Sequence" zu optimieren um auch mit diesem Gerät sinnvoll arbeiten zu können.

Dennoch ist das Prinzip der User-Startup nicht verkehrt. Vor allem die Tendenz, daß Anwendungsprogramme, die unbedingt gewisse Befehle in die „Startup-Sequence" packen müssen, diese in die User-Startup schreiben, ist sehr begrüßenswert. Denn so kann leicht unterschieden werden zwischen Kommandos, die für das System nötig sind und solche die von Applikationen benötigt werden. Die ist gerade in Hinsicht auf das spätere Entfernen einer Anwendung von der Festplette sehr hilfreich.

In Zeile 63 ist das Ende bereits zum greifen nahe: Der „rexxmast" wird gestartet - was er übrigens unter WB 2.1 nicht mehr wird. In Zeile 65 erfolgt dann das Laden der Workbench und in Zeile 67 gibt sich die Shell mit „endcli >nil:" schließlich selbst die Kugel. Damit ist dieses häßliche Kommando-Window weg, das vielleicht noch daran erinnern könnte, daß man Amigas auch mit der Tastatur bedienen kann. Beachten Sie das „>nil:" das vorhanden sein muß, damit das Fenster ordnungsgemäß geschlossen werden kann. Sollten Sie doch noch ab und zu (oder vielleicht gar immer) mit der Shell arbeiten wollen, dann können Sie diesen Befehl entfernen oder bevor Sie das ursprüngliche Window zur Hölle jagen, ein anderes, daß in Format und Eigenschaften Ihren Wünschen gerecht wird, öffnen.

6.4.2 Platz schaffen im Shell

Falls Sie beabsichtigen sollten, des öfteren in der Shell zu arbeiten, so können Sie sich eine spezielle Shell-Diskette anlegen, die ganz vorzüglich von unnötigen Programmen gereinigt werden kann. Dabei geht das Löschen überflüssiger Files noch viel weiter als bei den Aufräumarbeiten auf Workbench-Ebene (siehe: 5.2.1 Platz schaffen auf der Workbench). Sie sollten die folgenden Änderungen aber nicht an einer Diskette

vornehmen, die weiterhin als Workbench-Diskette verwendet werden soll, da dann einiges nicht mehr funktionieren würde. Den reinen Shell-Anwender wird dies allerdings wenig stören und um tieferen Einblick in den Aufbau der Workbench-Diskette zu bekommen, führt am Shell kein Weg vorbei.

Sehen wir uns zunächst einmal das Hauptverzeichnis an. An Files werden Sie hier nur „.info"-Dateien vorfinden, die für die Shell völlig uninteressant sind. Deshalb könnten zunächst alle Files mit der Endung „.info" gelöscht werden, nicht nur die im Hauptverzeichnis. Lediglich das Expansion-Directory ist eine Ausnahme, denn manche Treiber benötigen zum korrekten Laden die zugehörige „.info"-Datei.

Wenn man bei den Verzeichnissen alphabetisch vorgeht, kommen wir zuerst zum „C"-Directory. Dieser emthält den größten Teil der Shell-Kommandos. Daß auch hier einige Programme enthalten sind, die Sie nicht benötigen werden, versteht sich von selbst. So ist beispielsweise das Kommando „IconX" nur im Zusammenhang mit der Workbench sinnvoll einsetzbar. Doch welche weiteren Programme Sie hier löschen können, hängt von Ihnen an. Allgemein sollte man sich in diesem Verzeichnis etwas zurückhalten, denn die Programme sind eigentlich größtenteils sehr kurz und die meisten werden früher oder später benötigt. Greifen Sie hier also nur bei extremen Platzmangel zum „delete".
Im „Devs"-Verzeichnis finden Sie eine Reihe von Devices, die Mountlist, die System-Configuration und die Unterverzeichnisse Keymaps und Printers vor. Daß Sie nicht benötigte Druckertreiber und Tastaturbelegungen löschen können versteht sich von selbst. Die Devices an sich aber sind äußerst wichtig und gehören ebenso wie die Libraries wie die Butter aufs Brot.

Im „Expansion"-Directory befinden sich auf der Origialdiskette keine Files. Wenn ein Anwendungsprogramm oder die Installationsdiskette einer Hardwareerweiterung etwas hineinkopiert, wird es wohl auch benötigt werden, weshalb Sie nur dann etwas löschen sollten, wenn Sie ein komplettes Softwarepaket wieder entfernen.

Das „Fonts"-Verzeichnis ist das erste, in dem Sie gnadenlos zuschlagen können. Da die meisten Programme mit dem Standardzeichensatz arbeiten, der im ROM enthalten ist, werden zusätzliche Fonts nur selten benötigt. Da Sie im Laufe der Zeit sicherlich eine Reihe von Fonts ansammeln werden, ist es ohnehin eine gute Idee, sich eine spezielle Fonts-Diskette anzulegen. Dazu nehmen Sie einfach eine formatierte Diskette und kopieren alle Zeichensätze mit ihren Unterverzeichnissen in das Hauptverzeichnis dieser Diskette. Dann benennen Sie die Diskette in „Fonts" um. Wenn Sie nun auf einen der Zeichensätze dieser Diskette zugreifen wollen, legen Sie diese in ein freies Laufwerk. Was nun passiert ist folgendes: Da auf das Laufwerk entweder mit „DFx:" zugegriffen werden kann, oder aber mit dem Namen der Diskette, folgt in diesem Fall, daß die Disk mit „Fonts:" angesprochen wird - was genau dem Directory entspricht, in dem üblicherweise nach Zeichensätzen gesucht wird. Das zugewiesene „Fonts:"-Verzeichnis, im Normalfall „Workbench:Fonts", wird einfach übergangen. Mit diesem Trick können Sie auch mehrere „Fonts"-Disketten verwenden.

Auch im „L"-Verzeichnis könnte ein wenig gekürzt werden, z.B. der „speak-handler". Dann dürfte das Speak:-Device allerdings auch nicht mehr gemountet werden. Da die Handler aber nur wenige KBytes bringen, sollte hier nicht gelöscht werden um sich später nicht über unerklärliche Fehler den Kopf zerbrechen zu müssen.

Das „Libs"-Directory ist heilig. Löschen Sie keine Libraries, es ist für den Anwender nicht erkennbar, welche Libraries von einem Programm benötigt werden. Alle Libraries der Workbench könnten ohne Vorwarnung benötigt werden.

Das nächste Verzeichnis in dem gekürzt werden kann ist das „Prefs"-Directory. Die Preferences-Programme können alle gelöscht werden, wenn Sie Ihre Einstellungen erst einmal durchgeführt haben. Die Settings werden in dem Unterverzeichnis „Env-Archive" abgespeichert. Sollten Sie also einmal Ihre Einstellungen von eine auf eine andere Diskette übertragen wollen, dann müssen Sie nur diesen Schub kopieren. Im „Rexxc"-Directory befinden sich die ARexx unterstützende Programme. Da keines von ihnen die KByte-Grenze erreicht, verschwenden wir hier unsere Zeit.

Auch das „S"-Verzeichnis gibt nicht allzuviel her, doch sind hier einige Files völlig unbrauchbar. Lediglich die „Startup-Sequence" ist unverzichtbar. Falls inzwischen eine „User-Startup" erzeugt wurde, sollten Sie auch diese behalten. Und dann ist da noch die „Shell-Startup", die beim Öffnen einer neuen Shell aufgerufen wird. Alle anderen Skript-Dateien sind mehr oder minder entbehrlich. Allerdings sollten Sie daran denken, die „Startup-Sequence" entsprechend abzuändern, falls Sie „PickMap" löschen.

Der „System"-Ordner ist für den Shell-Anwender reichlich überflüssig. Programme, die Sie von hier benötigen können Sie auch gleich ins „C"-Directory kopieren. Damit entfällt auch der entsprechende Teil in der „Path"-Anweisung der „Startup-Sequence". Dies spart zudem noch Zeit, da ein Verzeichnis weniger durchsucht werden muß. Dasselbe gilt auch für das „Utilities"-Verzeichnis.

Was bleibt ist das „WBStartup"-Directory, dessen Bedeutung in der Shell Sie sich sicher vorstellen können: überhaupt keine. Folglich können Sie auch dieses Verzeichnis entfernen. Damit bekommen Sie allmählich ein freies Hauptverzeichnis, in dem sich leichter der Überblick behalten läßt. Sie können selbstverständlich noch nach und nach einiges mehr entfernen, jedoch besteht dann die Gefahr, daß einiges nicht mehr so läuft wie es soll. Sie sollten dennoch genügend Platz geschaffen haben um nun sinnvoll arbeiten zu können.

Teil II: Programmierung von AmigaOS 2.0

Nicht, daß Sie denken, nun würde ein kompletter Programmierkurs auf Sie warten. Doch wenn schon die ganze Zeit von neuen Funktionen und Features die Rede ist, so soll doch wenigstens kurz auf die Hintergründe eingegangen werden. Dieser Teil ist sicherlich nichts für den reinen Anwender, der keiner Programmiersprache mächtig ist, doch wer ist das auf dem Amiga-Sektor schon? Und Grundkenntnisse reichen völlig aus, um an dieser Stelle hinter den Vorhang zu blicken und so vielleicht ein besseres Verständnis davon zu bekommen, was in seinem Computer abläuft. Denn Sie müssen kein Automechaniker sein, um die Scheibenwischer auszuwechseln, doch wohl jeder weiß, was er zu tun hat, wenn er plötzlich nicht mehr durch die Windschutzscheiben sehen kann.

1. Neues

Bei der Betrachtung von AmigaOS 2.0 vom Standpunkt des Programmierers stechen einige Neuerungen ins Auge, die einem die Arbeit erheblich erleichtern. Doch auch an

Flexibilität hat das neue Betriebssystem gewonnen. Und das wichtigste: Über all diese Verbesserungen wurde nicht der Grund für den ganzen Aufwand, der Anwender, vergessen.

Zu berichten gibt es von neuen Libraries, die es zum Kinderspiel werden lassen, das professionelle Aussehen der Workbench 2.0 zu programmieren. Auch die bereits vorhandenen Bibliotheken wurden erweitert um die neuen Anwenderfunktionen in den Griff zu bekommen. Doch nicht nur die Anzahl der Betriebssystemfunktionen ist insgesamt gewachsen. Die Struktur des Betriebssystems selbst hat sich gewandelt, um den Anforderungen modernen Software-Designs zu genügen.

1.1 TagItems

Die wohl auffälligste Änderung in dieser Hinsicht sind die „TagItems". War man es bisher gewohnt, beim Funktionsaufruf Parameter entweder direkt oder in einer fest vorgegebenen Struktur zu übergeben, so kann dies nun flexibler erfolgen. Es hat sich nämlich herausgestellt, daß eine der wesentlichen Beschränkungen bei der Erweiterung der Systemfunktionen in den Strukturen liegt. Diese sind einfach nicht genügend erweiterbar um all die neuen Parameter aufzunehmen, die zum Zeitpunkt der Entwicklung von AmigaOS 1.2 noch nicht abzusehen waren. Deshalb wird teilweise noch mit erweiterten und ergänzten Strukturen, viel öfter aber mit „TagItems" gearbeitet. Das Vorgehen ist dabei folgendes: Anstatt bestimmte Parameter an festgelegten Positionen in einer Struktur zu erwarten, wird ein Feld von variabler Länge übergeben. In diesem sind dann jeweils ein Kennwort („Tag") und dahinter der eigentliche Parameter aufgeführt. Diese Kombination kann beliebig oft aufeinanderfolgen, so daß

die Anzahl der Parameter unbegrenzt bleibt. Das Ende des Feldes wird ebenfalls durch ein besonderes „Tag" gekennzeichnet. Der Vorteil ist klar ersichtlich: Wenn in neueren Betriebssystemversionen zusätzliche Parameter zur Verfügung stehen, brauchen lediglich neue „Tags" bekanntgegeben werden. Sind diese bei einem Funktionsaufruf nicht mit angeführt, so werden Standardwerte herangezogen, ansonsten die gegebenen. Es ist nicht mehr nötig, die Strukturen immer wieder auf Neue zu erweitern.

Doch auch jetzt haben Sie mit dieser Vorgehensweise einen enormen Vorteil: Sie müssen nicht mehr eine unter Umständen sehr lange Struktur mit vielen Parametern füllen, sondern übergeben nur noch die Werte, die vom Standard abweichen. Es ist - zumindest zu dieser Version von AmigaOS - bei älteren Funktionen aber auch immer noch möglich, die alten Strukturen zu übergeben, oder auch beides, wobei die „Tags" dann Vorrang haben. Dieses Vorgehen ist natürlich recht sinnlos; entweder oder!
Jedoch sollte man bedenken, daß vielleicht einmal das Konzept der Strukturen ganz fallen gelassen wird und dann nur noch „TagItems" den Laden schmeissen. Bis dahin wird aber mit Sicherheit noch einige Zeit vergehen, damit die Kompatibilität gewährleistet bleibt. Wer allerdings Programme schreibt, die schon jetzt derart intensiven Gebrauch von den neuen Funktionen machen, daß sie ohne Amiga OS2.0 ohnehin nicht laufen, der sollte auch voll und ganz auf das Konzept der „TagItems" umsteigen. Es mag am Anfang etwas ungewohnt sein, doch es lohnt sich auf jeden Fall.

1.2 User Interface Style

Weniger mit dem Programmieren selbst als mit dem Programmdesign hat der Begriff „User Interface Style" zu tun. Hier geht es um die Form der Darstellung von Informationen und der Konzeption der Benutzerschnittstelle. Um ein Besipiel zu nennen: Beim Apple Macintosh wurden von Anfang an zwingende Konventionen festgelegt, wie ein Programm auszusehen hat. Dazu gehörte der Aufbau von Menüs, wo welche Gadgets zu stehen haben und vieles mehr. Die Folge davon ist, daß ein Anwender sich nur in ein einziges Programm einarbeiten muß und sich sofort im nächsten zurechtfindet. Natürlich werden immer andere Funktionen geboten, doch sehen sie einfach immer alle gleich aus, es existiert ein einheitlicher „User Interface Style". Und wie förderlich solche Konventionen sind, die den Lernaufwand senken und damit die Produktivität steigern, hat auch Commodore inzwischen eingesehen. Die folge ist, daß zu AmigaOS 2.0 erstmals genauere Forderungen aufgestellt wurden. Ob und wenn ja, in welchem Maße diese nun eingehalten werden, hängt letztendlich von den Programmierern ab. Doch als Anreiz zur Realisierung ist neben der zu erwartenden höheren Akzeptanz bei der Kundschaft ein noch viel wesentlicherer Aspekt gegeben: Die Unterstützung auf Programmierebene.

Das wohl beste Beispiel hierfür sind die Filerequester. Unter AmigaOS 1.2/1.3 wurden keine Funktionen für einen Filerequester zur Verfügung gestellt und jeder mußte seinen eigenen erstellen. Dabei hatten die verschiedenen Programmierer natürlich andere Vorstellungen von dem Aussehen ihres Filerequester. Und die Folge davon war, daß der Anwender bei jedem neuen Programm immer wieder nach der „Parent"- oder „Cancel"-Funktion suchen mußte. Noch schlimmer war, daß einige Filerequester

dringend benötigte Funktionen gar nicht erst anboten oder manche Programme erst gar keine Fileselektion anboten, sondern der Filename von Hand eingetippt werden mußte. Die ersten Ansätze einer Rettung boten die „arp.library" und die „req.library" die beide geeignete Funktionen boten und so wenigstens teilweise zu einer Vereinheitlichung dienten.

Doch jetzt ist alles anders! Die „asl.library" die bei der Workbench 2.0 fest zum Lieferumfang gehört steht endlich jedem Programmierer zur Verfügung und sie löst das Problem der Filerequester auf geradezu vorbildliche Weise. Die exakte Definition, wie ein solcher Requester auszusehen hat, würde nur die wenigsten interessieren, wenn sie ihn doch wieder selbst machen müssten, da viele Programmierer auf ihre alten Routinen zurückgreifen würden. Doch mit dem Angebot, ohne viel Aufwand in den Konventionen bleiben zu können, steht der Realisierung des geforderten „User Interface Styles" nichts mehr im Wege.

Im Grunde genügt es, wenn Sie sich den Aufbau der Workbench-Programme ansehen, um zu wissen, wie Sie Ihre eigenen Programme gestalten sollten. Wenn Sie es aber genauer wissen möchten seien Sie auf das Buch „User Interface Style Guide" verwiesen. Es beschreibt die Forderungen äußerst ausführlich und kann wirklich jede Unklarheit beseitigen. Normalerweise ist es aber nicht unbedingt notwendig, sich so kleinlich an die gestellten Forderungen zu halten; man sollte aber zumindest problemlos mit Ihrem Programm zurechtkommen, wenn man den Standard kennt.

2. Neue Funktionen

Die für den Programmierer wohl auffälligste Veränderung sind die neuen Libraries sowie die Erweiterungen der alten. Dadurch stehen eine Unzahl neuer Funktionen zur Verfügung, mit denen sich geradezu Fantastisches anfangen läßt. Natürlich kann hier keine wirklich umfassende Beschreibung erfolgen, doch Sie sollen wenigstens auf den Geschmack kommen.
Deshalb wurden nur einige wenige Funktionen ausgewählt, sozusagen die Schmankerl! Und zumindest diese werden ausreichend dargestellt, so daß Sie mit den gegebenen Informationen sofort auf die wesentlichsten Neuerungen zugreifen können.

Zum Format, in dem beispielsweise die Parameter aufgeführt sind, ist zu sagen, daß den üblichen Konventionen entsprechend sowohl die Angaben, die für die Realisierung in der Sprache „C" nötig sind, wie auch die Register aufgeführt sind, so daß auch Assembler-Programmierer sofort loslegen können. Um Ihr Programm auch korrekt übersetzen lassen zu können, benötigen Sie nur noch die Include-Files zu AmigaOS 2.0, die Sie sicherlich als Update zu Ihrem Compiler / Assembler erwerbern können.

2.1 DOS-Library

Die DOS-Library befaßt sich mit allen Arten von File-Operationen. Dazu gehört seltsamerweise auch die Ein-/Ausgabe von Texten in Shell-Fenstern. Da diese jedoch auch von bzw. in Files umgelenkt werden können, ist dies gar nicht allzu verwunderlich. Bisher jedoch war man gerade auf diesem Gebiet recht allein gelassen, was sich nun

glücklicherweise geändert hat. Und auch die wirklichen File-Zugriffe sind jetzt so einfach zu realisieren wie in BASIC.

2.1.1 ReadArgs

„ReadArgs" untersucht die Kommandozeile.
result = ReadArgs(template, array, rdargs)
D0 D1 D2 D3

struct RDArgs * ReadArgs(STRPTR, LONG *,struct RDArgs *)

Beim Aufruf eines jeden Programms können Sie Parameter angeben, indem Sie diese hinter dem Namen des Programms in der selben Befehlszeile anfügen. Ob das Programm diese dann auch verarbeitet, ist eine andere Frage. Die bisherige Vorgehensweise lag darin, die Zeile, deren Adresse im Speicher in A0 (bzw. Argv) und deren Länge in D0 (bzw. Argc) übergeben wurde, nach Parametern zu durchsuchen und selbst zu entscheiden, ob diese in Anzahl und Format mit dem übereinstimmen, was gefordert wurde.
Mit „ReadArgs" geht das ganz anders. Vergessen Sie A0 und D0! Alle Arbeit wird Ihnen abgenommen. Sie brauchen nichts weiter zu tun, als ein „template" zu erzeugen, was nichts anderes ist, als die Formatschablone für die Parameter. Das ist genau das Ding, das Sie angezeigt bekommen, wenn Sie ein Shell-Kommando mit dem Parameter „?" aufrufen. Da erscheint dann beispielsweise „Filename/A,Quiet/S" oder ähnliches (was die Buchstaben hinter dem Schrägstrich bedeuten, wurde bereits im Shell-Teil, 6.3.1 Parameterschablone, erklärt). Damit wird ausgedrückt, daß maximal zwei Parameter

erwartet werden. Für diese muß ausreichend Speicherplatz, d.h. zwei Langwörter (für jeden Parameter eins) reserviert und ein Zeiger darauf als „array" übergeben werden. „rdargs" brauchen Sie im Normallfall nicht zu übergeben, denn dann wird diese automatisch für Sie allokiert. Wenn Sie jedoch bereits eine vorgefertigte haben (was ziemlich unwahrscheinlich ist, da Sie ja wohl nur ein einziges Mal in Ihrem Programm die Parameter auslesen), können Sie diese hier angeben.
Man beachte aber, daß diese Struktur mit der Funktion „AllocDosObject" allokiert werden sollte. Nach dem Aufruf erhalten Sie als „result" entweder die gefüllte „RDArgs"-Struktur zurück, oder aber Null, falls ein Fehler aufgetreten ist. Fehlerquellen können z.B. die falsche Anzahl von Parametern, ein nicht vorhandenes Schlüsselwort oder aber auch der Aufruf des Programms mit „?" sein. Denn in diesem Fall erkennt „ReadArgs" automatisch, was der Anwender damit will und teilt Ihnen dann mit, daß das eigentliche Programm nicht gestartet werden braucht.

Der Aufbau der „RDArgs"-Struktur hängt nun von dem übergebenen „template" ab. Zwar wird für jedes Argument immer ein Langwort verbraucht, doch die Bedeutung dieses Langworts ist unterschiedlich. Im Normalfall stellt es einen Zeiger auf eine Zeichenkette da, die mit einem Nullbyte abgeschlossen wird. Wurde der Parameter in Anführungszeichen eingeschlossen eingegeben, um z.B. Leerzeichen korrekt zu übergeben, so wurden diese bereits entfernt.

Sollte jedoch eine Zahl als Parameter erwartet werden („/N"), so liegt diese bereits fertig in binär umgerechnet in dem Langwort. Bei Schaltern („/S") ist das entsprechende Langwort entweder Null (für nicht gesetzt) oder von Null verschieden (für gesetzt). Wenn es möglich war, zu einem Parameter mehrere Argumente zu übergeben („/M"),

so ist in dem Langwort ein Zeiger auf eine Tabelle gegeben, die wiederum Zeiger auf die eigentlichen Argumente enthält.
Sie sehen also, es ist nicht mehr nötig, die Parameterzeile aufwendig zu zerlegen, nach Schlüsselwörtern zu suchen und Zahlen umzurechnen - das geschieht alles automatisch! Und selbst wenn die Eingabe inkorrekt war, können Sie jegliche Verantwortung abwälzen, und das geht mit der folgenden Funktion:

2.1.2 PrintFault

„PrintFault" gibt den Text zu einem DOS-Fehlercode aus.
success = PrintFault(code, header)
D0 D1 D2

BOOL PrintFault(LONG, STRPTR)

Sie kennen sicherlich die seltsamen Fehlercodes die bei früheren Versionen von AmigaOS teilweise ausgegeben wurden, wie etwa „103" bei ungenügendem Speicherplatz. Damit der Anwender nicht alle Zahlenwerte auswenig lernen oder nachschlagen muß und der Programmierer nicht in jedem Programm von neuem alle Fehlertexte aufführen muß, gibt es „PrintFault". Sie übergeben dieser Funktion lediglich den Fehlercode als „code" und können darüber hinaus noch einen eigenen Text vor der eigentlichen Fehlermeldung ausgeben, wozu Sie einen Zeiger darauf in „header" bereitstellen. Der Text wird auf dem Standard-Ausgabekanal ausgegeben, mit anderen Worten in das Shell, von dem aus das Programm gestartet wurde. Ein vorheriger Aufruf von „Output" entfällt.

Den Fehlercode können Sie entweder selbst liefern, beispielsweise schreiben sie 103 in „code" wenn ein „AllocMem" fehlschlug, oder in Verbindung mit DOS-Funktionen vorher erfragen. Dies läuft über die „IoErr" Funktion, die keine Parameter hat. Sie liefert immer dann den richtigen Fehlercode, wenn die zuvor aufgerufene DOS-Funktion einen Fehler meldete.
Um das Vorgehen im Zusammenhang mit „ReadArgs" klar zu machen, folgendes kurze Beispiel:

```
move.l #template,d1
move.l #array,d2
moveq  #0,d3
callsys ReadArgs
tst.l  d0
bne    allesOK

callsys IoErr
move.l d0,d1
moveq  #0,d2
callsys PrintFault
bra    exit
```

Das war auch schon alles. Bei „exit" geben sie ggf. vorher allokierten Speicher frei und schließen geöffnete Bibliotheken und kehren mit einem geeigneten Returncode zurück. Falls der Anwender Ihr Programm nun mit den falschen Parametern oder mit „?" aufruft, erledigen diese paar Zeilen alles, was zu einer korrekten Fehlerbehandlung

nötig ist. Und das beste kommt erst noch: Sollten Sie bereits die Workbench 2.1 benutzen, erfolgen die Ausgaben sogar noch in der gewählten Landessprache. Dann heißt es eben „Speicherplatzmangel", der Anwender weiß, was los ist und den Programmierer kostet es kein einziges graues Haar.

2.1.3 ErrorReport

„ErrorReport" zeigt einen Retry/Cancel Requester zu einem Fehler an.
status = ErrorReport(code, type, arg1, device)
D0 D1 D2 D3 A0

BOOL ErrorReport(LONG, LONG, ULONG, struct MsgPort *)

Nicht immer ist es wünschenswert, wenn ein Programm beim Auftritt eines Fehlers gleich abbricht. Oftmals wäre es viel sinnvoller, dem Anwender noch einmal die Gelegenheit zu geben, die Sache in Ordnung zu bringen, und dazu gibt es ja Requester. Unter AmigaOS 2.0 braucht der Programmierer nun nicht mehr selbst von Hand eine Requester-Struktur aufzubauen, sondern er findet im Rahmen der DOS-Funktionen genau das was er braucht: „ErrorReport".

Sie übergeben zu diesem Zweck in „code" den entsprechenden Fehlercode, der der Anlaß für den Requester ist. Zur Auswahl stehen zur Zeit beispielsweise „Disk not validated", „Disk write protected" oder „Disk full". Als „arg1" muß irgendeine Form von Verweis auf das betroffene Device vorliegen, wobei Sie relativ frei sind. Was hier übergeben wird, muß lediglich in „type" angegeben werden, wobei folgende Auswahl

besteht: Es handelt sich um einen Lock, ein Filehandle, eine Volume oder einen String des Namens der Volume. Wenn Sie also ein File öffen und sein Filehandle erhalten, sich anschließend aber herausstellt, daß ein Schreibschutz besteht, so können Sie direkt das Filehandle übergeben. Haben Sie nur eine der anderen Formen zur Verfügung, brauchen Sie sich nicht erst ein Filehandle besorgen, sondern übergeben einfach diese Daten.
„Device" wird praktisch nie benötigt. Die einzige Ausnahme ist, wenn Sie als „type" einen Lock angeben und „arg1" auf Null setzen. Dann wird für „device" die Adresse das Handler-Tasks erwartet, wodurch die Anzahl der möglichen Typen effektiv nochmals um eins erhöht wird.
Nach dem Aufruf dieser Funktion wird dann ein hübscher Requester erscheinen und Ihr Problem verkünden und auf eine Entscheidung des Anwenders warten. Sollte es sinnvoll sein, wird das System auch das Einlegen einer Diskette als Auslöser für „Retry" wählen, wie man es ja auch von den Standard-Requestern gewöhnt ist. Der Rückgabewert ist Null für „Retry", ungleich Null für „Cancel" oder einen aufgetretenen Fehler.

2.1.4 PutStr

„PutStr" gibt einen String auf dem aktuellen Ausgabekanal aus.
error = PutStr(str)
D0 D1

LONG PutStr(STRPTR)

Doch nicht nur Fehlermeldungen können komfortabel ausgegeben werden. Es kommt häufig genug vor, daß einfach nur ein kurzer Text ausgegeben werden muß, der den Anwender über irgendwelche Vorgänge informiert. Bisher war das Vorgehen zu diesem Zweck, erst den Ausgabekanal zu bestimmen und dann mit der „Write" Funktion in diesen zu „schreiben". Dazu mußte dann auch noch die Länge der Zeichenkette bestimmt werden und überhaupt war die ganze Sache sehr unangenehm.
Mit „PutStr" hat sich das geändert. Sie haben hier die Minimalform eines „Print"-Befehls vor sich. Sie übergeben nichts weiter als einen Zeiger auf den auszugebenden String, der mit einem Nullbyte terminiert ist, in „str". Die Zeichenkette wird dann ausgegeben und auch noch mit einem Newline versehen, so daß für die nächste Ausgabe der Cursor bereits in der nächsten Zeile steht. Einfacher geht es wirklich nicht mehr. In „error" wird noch mitgeteilt, ob ein Fehler aufgetreten ist, was durch den Wert -1 angezeigt wird. Ist kein weiteres Problem aufgetaucht, erhalten Sie Null zurück.

2.1.5 VPrintf

„VPrintf" formatiert einen String und gibt diesen aus.
count = VPrintf(fmt, argv)
D0 D1 D2

LONG VPrintf(STRPTR, LONG *)

Wenn „PutStr" die Minimalform eines Print-Befehls war, dann haben Sie mit „VPrintf" die vollständige Variante. Oder besser gesagt, „VPrintf" ist die Betriebssystem-Version der C-Funktion „printf". Wieder einmal liegt der Vorteil auf der Hand. Die Funktion muß nicht zu dem Programm gelink t werden und spart folglich Platz,

außerdem ist ein Zeitgewinn zu erwarten. Und der Assembler-Freak, der ja auf keine C-Funktion zurückgreifen kann, erhält eine Äquivalente.
Was Sie übergeben müssen, ist lediglich ein Zeiger auf die auszugebende Zeichenkette, in der auch die Formatzeichen enthalten sind, in „fmt". In „argv" muß außerdem noch der Zeiger auf das Array stehen, in dem die einzufügenden Daten aufgeführt sind. Das können entweder Zahlenwerte oder wieder Zeiger auf Zeichenketten sein. Die Reihenfolge der Werte korrespondiert mit dem Auftreten im Formatstring. Der Aufbau eines Formatzeichens ist wie folgt:

%[flags][width.limit][length]type

Mit anderen Worten, ein Formatzeichen wird immer durch ein „%" eingeleitet. Sollten Sie das Prozentzeichen selbst ausgeben wollen, so müssen Sie „%%" schreiben. Darauf folgen muß nur noch der „type", alle anderen Parameter sind optional. Sie bedeuten:

flags
nur „-" als Parameter möglich; bedeutet, daß linksbündig ausgegeben wird

width
bestimmt die Breite der Ausgabe (für Tabellen sinnvoll); ist das erste Zeichen eine „0", so wird die gesamte Ausgabe von links her mit „0"en aufgefüllt; der Punkt („.") muß folgen, falls „width" angegeben wird

limit
die maximale Anzahl von Zeichen, die ausgegeben werden; nur für „%s" (Strings) gültig

length
die Größe eines Zahlenwertes; „l" bedeutet Langwort, wenn „length" nicht angegeben wird, bedeutet das „Wort"

Abhängig vom „type" wird der Wert aus dem Array entsprechend interpretiert:
b
ein BPTR-String

d
eine Dezimalzahl

u
eine vorzeichenlose Dezimalzahl

x
eine Hexadezimalzahl

s
ein String (Null-terminiert)

c
ein Zeichen

Einige Besipiele für korrekte Formatsrings sollten die letzten Unklarheiten beseitigen:

fmt : „Die Höchstgeschwindigkeit beträgt %ld %s!"
argv: 100, „km/h"
=> : Die Höchstgeschwindigkeit beträgt 100 km/h!

fmt : „Speicher allokiert bei Adresse 0x%lx."
argv: $4ef30
=> : Speicher allokiert bei Adresse 0x4ef30.

fmt : „%ld +%ld%% MwSt = %6.ld"
argv: 100, 14, 114
=> : 100 + 14% MwSt = 114

2.1.6 FPutC

„FPutC" gibt ein einziges Zeichen auf dem spezifizierten Ausgabekanal aus.
char = FPutC(fh, char)

D0 D1 D2
LONG FPutC(BPTR, UBYTE)

Die Hauptaufgabe eines DOS ist das Lesen und Schreiben von Daten von bzw. in Files. „FPutC" ist die einfachste der schreibenden Funktionen. Sie übergeben lediglich in „fh" ein Filehandle, wie Sie es von „Open" geliefert bekommen haben und in „char" das Byte, das ausgegeben werden soll. Der Rückgabewert „char" enthält dann entweder wieder dieses Zeichen oder EOF, falls ein Fehler aufgetreten ist.
Diese Ausgabe erfolgt gepuffert, was heißt, daß nicht für jedes Byte das Laufwerk angeworfen wird, sondern erst alle Ausgaben in einen Puffer wandern. Erst bei einem Newline, Return oder Nullbye wird der Puffer tatsächlich geschrieben. Dieses Schreiben läßt sich allerdings auch erzwingen (mit der Funktion „Flush") oder dem Schließen des Files.

2.1.7 FPuts

„FPuts" gibt einen String auf dem spezifizierten Ausgabekanal aus.
error = FPuts(fh, str)
D0 D1 D2

LONG FPuts(BPTR, STRPTR)

Diese Funktion ist „FPutC" sehr ähnlich, jedoch wird nicht nur ein einziges Byte, sondern eine ganze Zeichenkette ausgegeben. Diese muß mit einem Nullbyte abgeschlossen sein, das nicht mehr mit ausgegeben wird. Als „error" bekommen Sie

entweder Null zurückgeliefert, wenn alles glatt ging, oder -1 beim Auftreten eines Fehlers (was den üblichen Konventionen im DOS entgegengesetzt ist!). Auch diese Ausgabe erfolgt gepuffert.
Diese Funktion macht es gerade dem Assembler-Programmierer sehr leicht, wie in einer Hochsprache zu programmieren. Und der Programmierer einer Hochsprache bekommt einen Ersatz für seine üblichen Ausgabefunktionen geliefert, die auf Systemebene läuft und daher weit schneller ist. Denn was ist denn FPuts? Doch nichts anderes als das bekannte „Print#" aus BASIC. Beim Herumhantieren mit Textfiles kann diese Funktion hervorragend dazu verwendet werden, eine einzige Zeile zu schreiben. Sie müssen lediglich die einzelnen Zeilen so im Speicher anordnen, daß sie immer Null-terminiert sind.

2.1.8 FGetC

„FGetC" liest ein einziges Zeichen von dem spezifizierten Eingabekanal ein.
char = FGetC(fh)
D0 D1

LONG FGetC(BPTR)

Das Gegenstück zu „FPutC" ist „FGetC". Es wird ein einzelnes Zeichen aus dem File, das durch sein Filehandle in „fh" spezifiziert wurde. Falls ein Fehler auftreten sollte, oder das Fileende erreicht wurde, wird -1 als „char" zurückgegeben, ansonsten das gelesene Byte. Um letztendlich zu entscheiden, ob tatsächlich ein Fehler aufgetreten ist, kann „IoErr" aufgerufen werden. Natürlich ist auch diese Eingabe gepuffert.

2.1.9 FGetS

„FGets" liest einen String von dem spezifizierten Eingabekanal ein.
buffer = FGets(fh, buf, len)
D0 D1 D2 D3

STRPTR FGets(BPTR, STRPTR, ULONG)

„FGets", das Gegenstück zu „FPuts" liest einen String vom in „fh" angegebenen Filehandle ein. Dabei bedeutet „String", daß so viele Bytes gelesen werden, bis ein Newline oder das Ende des Files erreicht wurden, wobei das Newline dann noch mitgelesen wird. Maximal aber werden „len"-1 Bytes gelesen. Das „-1" ist deshalb wichtig, damit das letzte Byte auf Null bleibt und mit „FPuts" korrekt geschrieben werden kann. Als „buf" muß selbstverständlich ein Zeiger auf einen ausreichend (d.h. von der Größe „len") dreimensionierten Puffer übergeben werden, der dann gefüllt wird. Der Rückgabewert „buffer" ist dann entweder wieder dieser Zeiger oder Null, was bedeutet, daß ein Fehler aufgetreten ist, bzw. das Fileende erreicht wurde, ohne daß vorher noch Zeichen hätten gelesen werden können. Die Entscheidung, welcher der beiden Fälle eingetreten ist, trifft man über „IoErr". Auch „FGets" ist gepuffert.

Diese Funktion entspricht also einem „Input#" von BASIC. Damit wird es zum Kinderspiel, Texte auf Zeilen formatiert einzulesen. So lassen sich beispielsweise Konfigurationsfiles, die im ASCII-Format vorliegen sehr speicherplatzsparend bearbeiten. Sie allokieren einen ausreichenden Puffer (also mindestens 81 Bytes, was einer Zeile + einem Nullbyte entspricht) und rufen „FGets" auf. Die gelesene Zeile

werten Sie aus und lesen anschließend in den selben Puffer die nächste Zeile ein. Bisher bestand das Problem darin, daß man entweder das gesamte File einlesen mußte oder nie genau eine Zeile traf, weil deren Länge vorher ja nicht bekannt war. Nun ist das alles kein Problem mehr. Und sollte der Puffer wirklich einmal zu klein sein, muß eben abhängig von der jeweiligen Anwendung reagiert werden und möglicherweise zusätzlicher Speicher angefordert werden. Da diese Funktion aber auf Textfiles zugeschnitten ist, die im Normalfall die Länge einer Zeile nicht überschreiten, hat man hier einen sehr guten Anhaltspunkt für die Puffergröße.

2.1.10 ExamineFH

„ExamineFH" erfragt Informationen zu einem geöffneten File.
success = ExamineFH(fh, fib)
D0 D1 D2

BOOL ExamineFH(BPTR, struct FileInfoBlock *)

Mit „ExamineFH" wird nun ein ganz anderes Thema angesprochen, das aber nicht minder bemerkenswert ist. Es geht um Filehandles und Locks. Die einen erhält man beim Öffnen von Files, die anderen lassen sich mit einer eigenen Funktion („Lock") erstellen und können sich auch auf Verzeichnisse beziehen. Zwei Strukturen also, die sich sehr ähnlich sind und über die man beide auf Files zugreifen kann - jedoch immer nur auf gewisse Bereiche. So war es zumindest bisher: Wenn Sie Daten lesen oder schreiben wollten, war ein Filehandle nötig, zum Auslesen des Comments oder zur Abrage der Filegröße ein Lock. Bei AmigaDOS 2.0 hat sich die Situation stark

verändert, es wurden viele neue Funktionen aufgenommen, die dasselbe tun, wie die ihnen entsprechenden alten, aber die jeweils andere Struktur als Parameter annehmen.
„ExamineFH" soll nun ein Repräsentant für die Vielzahl dieser neuen Funktionen sein, der ganz besonders nützlich ist. Das Problem, das sich nun so galant lösen läßt, ist folgendes: Sie möchten ein File komplett in den Speicher lesen. Eigentlich keine große Sache, doch wenn Sie die Länge des Files nicht von vornherein angeben können, muß diese zunächst bestimmt werden, um auch genügend Speicher zur Verfügung stellen zu können. Dazu war bisher immer erst ein Lock nötig, um dann mittels „Examine" den FileInfoBlock zu besorgen und aus diesem die Filelänge auszulesen. Anschließend wird dann das File geöffnet und gelesen.

Doch mit „ExamineFH" gehen Sie anders vor: Zuerst das File öffnen und das erhaltene Filehandle als „fh" dann „ExamineFH" übergeben. Weiterhin muß noch ein leerer FileInfoBlock in „fib" bereitgestellt werden, also mit anderen Worten ein Speicherblock der entsprechenden Größe. Nach dem Aufruf der Funktion können Sie die Filelänge auslesen und anschließend sofort mit dem bereits vorhandenen Filehandle das komplette File einlesen, falls „success" nicht auf einen Fehler hingewiesen hat. Zum Abschluß wird das File noch ordnungsgemäß geschlossen und Sie haben nur noch vier Funktionsaufrufe benötigt, nicht mehr sechs.

Zu beachten ist noch, daß es ja auch möglich ist, die Funktion „ExamineFH" nicht sofort nach dem Öffnen, sondern vielleicht nach einigen anderen Filezugriffen aufgerufen werden kann. Insbesondere kann sich seit dem Öffnen die Größe des Files verändert haben, z.B. wenn es inzwischen beschrieben wurde. In diesem Fall wird nicht

garantiert, daß die gelieferte Information dem aktuellen Stand der Dinge entspricht, wenngleich versucht wird, korrekte Werte zu liefern. Wenn Sie allerdings darauf achten, die Filelänge vor derartigen Änderungen auszulesen und selbst eventuelle Veränderungen zu verwalten, entstehen aus dieser Unzulänglichkeit keine weiteren Einschränkungen.

2.2 ASL-Library

Die ASL-Library, die die neuen File Requester zur Verfügung stellt, wartet mit nur sechs Funktionen auf. Das ist jedoch keineswegs ein Hinweis auf eine eingeschränkte Nutzbarbeit sondern vielmehr ein Ausdruck der enormen Mächtigkeit ihrer Funktionen. Und auch die Funktionen selbst sind so unglaublich einfach zu handhaben, daß es nun endlich keine Ausrede mehr gibt, keinen File Requester in ein Programm einzubauen. Wegen der Kürze dieser Library ist es sogar an dieser Stelle möglich, alle Funktionen umfassend genug zu beschreiben.

Ein wichtiger Hinweis gleich vorweg: Die ASL-Library, die ja erst seit AmigaOS 2.0 existiert, macht intensiven Gebrauch von TagItems. Das geht sogar so weit, daß es verboten ist, irgendwelche Werte in der Request-Datenstruktur anders als über TagItems zu ändern. Dies bedeutet nichts anderes, als daß diese Struktur nur gelesen, aber nicht beschrieben werden darf. Da die Tags in mehreren der folgenden Funktionen angegeben werden können und immer gleich sind, sollen diese gleich hier zu Beginn angegeben werden:

ASL_Hail
Text, der in der Titelzeile des Windows angezeigt wird

ASL_Window
Zeiger auf das Parent Window, also das Fenster des Programms, das den ASL-Requester aufruft (damit dieser auf dem richtigen Screen erscheint)

ASL_LeftEdge
Position des linken Rands des Requesters

ASL_TopEdge
Position des oberen Rands des Requesters

ASL_Width
Breite des Requesters

ASL_Height
Höhe des Requesters

ASL_HookFunc
Zeiger auf eine sog. „Hook"-Funktion; etwas schwierig zu handhaben und nur in den seltensten Fällen notwendig

ASL_File
Filename beim Aufruf

ASL_Dir
Pfadname beim Aufruf

ASL_FontName
Name des Fonts beim Aufruf (nur bei Font Requestern)

ASL_FontHeight
Höhe des Fonts beim Aufruf (nur bei Font Requestern)

ASL_FontStyle
Stil des Fonts beim Aufruf (nur bei Font Requestern)

ASL_FontFlags
Flags des Fonts beim Aufruf (nur bei Font Requestern)

ASL_FrontPen
Farbe des FrontPen (nur bei Font Requestern)

ASL_BackPen
Farbe des BackPen (nur bei Font Requestern)

ASL_MinHeight
minimale Höhe des Displays für Fonts (nur bei Font Requestern)

ASL_MaxHeight
maximale Höhe des Displays für Fonts (nur bei Font Requestern)

ASL_OkText
Text, der anstatt des „OK" angezeigt werden soll (max. sechs Zeichen)

ASL_CancelText
Text, der anstatt des „Cancel" angezeigt werden soll (max. sechs Zeichen)

ASL_FuncFlags
Function Flags, die vom Typ des Requesters anghängen, z.B. FILE_SAVE für File Requester

ASL_ExtFlags1
Extended Flags, z.B. FIL1F_NOFILES für File Requester

Wie bereits über TagItems allgemein gesagt, gilt auch hier, daß keiner der Tags angegeben sein muß. In diesem Fall werden Standardwerte genommen, die für die meisten Anwendungen korrekt sind. Je mehr Sie den Requester an Ihre Wünsche anpassen wollen, um so mehr Tags müssen Sie angeben, es geht aber auch ganz ohne.

2.2.1 AllocAslRequest

„AllocAslRequest" allokiert einen ASL-Requester, der mit TagItems modifiziert werden kann.

request = AllocAslRequest(type, ptags)
D0 D0 A0

APTR AllocAslRequest(ULONG, struct TagItem *)

Zu jedem ASL-Requester existiert eine Struktur, in der Informationen bezüglich Aussehen, Position, Typ und anderem enthalten sind. Auch das Ergebnis einer Auswahl aus dem Requester wird hier abgelegt und kann ausgelesen werden. Doch zuerst einmal muß eine solche Struktur erstellt werden und das geht mit „AllocAslRequest". Dabei wird nicht nur der nötige Speicherplatz besorgt, sondern auch gleich die Standardwerte eingefügt, die sich zudem mittels TagItems, die Sie in „ptags" übergeben können, modifizieren lassen. Außerdem wird beim Aufruf der Typ des Requesters ausgewählt; momentan stehen „ASL_FileRequest" (File Requester) und „ASL_FontRequest" zur Auswahl.

Als „result" bekommen Sie entweder einen Zeiger auf die entsprechende Struktur geliefert, oder Null, falls irgendein Fehler aufgetreten sein sollte. Diesen Zeiger verwenden Sie im Folgenden als Parameter für „AslRequest" und abschließend für „FreeAslRequest", um die Struktur wieder freizugeben.

2.2.2 AllocFileRequest

„AllocFileRequest" allokiert eine FileRequester Struktur.
request = AllocFileRequest()
D0

struct FileRequester *AllocFileRequest()

Wenn Sie nur einen ganz gewöhnlichen File Requester ohne irgendwelche Extras benötigen, geht das Ganze sogar noch einfacher: Die Funktion „AllocFileRequest" benötigt noch nicht einmal einen Parameter. Als „result" erhält man wie auch bei „AllocAslRequest" einen Zeiger auf eine FileRequester Struktur.

2.2.3 AslRequest

„AslRequest" erfragt mittels eines ASL Requesters eine Eingabe vom Anwender.
result = AslRequest(request, ptags)
D0 A0 A1

BOOL AslRequest(APTR, struct TagItem *)

Um nun endlich den Requester auf den Bildschirm zu zaubern, rufen Sie „AslReuqest" auf. Als Parameter wird die vorher allokierte Struktur in „request" übergeben, sowie weitere TagItems in „ptags", wenn Bedarf besteht. Dies ist entweder dann der Fall, wenn die Struktur mit „AllocFileRequest" erstellt wurde, da diese Funktion keine

TagItems als Parameter akzeptiert, oder wenn sich seit dem letzten Aufruf Änderungen ergeben haben, die nun nachträglich übermittelt werden müssen. Dabei ist es wichtig zu wissen, daß die aktuellen Werte auch bei erneutem Aufruf erhalten bleiben, solange sie nicht über TagItems geändert werden.
Das „result" wird nur dann Null, wenn ein Fehler auftrat oder der Anwender „Cancel" gewählt hat. Ansonsten können Sie das Ergebnis des Requesters aus der Struktur auslesen, die Sie übergeben hatten.

2.2.4 RequestFile

„RequestFile" erfragt mittels ASL Requesters eine Eingabe vom Anwender.
result = RequestFile(request)
D0 A0

BOOL RequestFile(struct FileRequester *)

Fast identisch zu „AslRequest" ist „RequestFile". Der einzige Unterschied besteht darin, daß bei „RequestFile" keine TagItems-Liste mehr angegeben werden kann. Deshalb seien Sie auch auf weitere Details, betreffend auf 2.2.3 AslRequest, verwiesen.

2.2.5 FreeAslRequest

„FreeAslRequest" gibt einen mit „AllocAslRequest" allokierten Requester frei.

FreeAslRequest(request)
A0

FreeAslRequest(APTR)

Um die Struktur und alle mit ihr verbundenen Resourcen wieder freizugeben, muß „FreeAslRequest" aufgerufen werden. Als einziger Parameter wird ein Zeiger auf diese Struktur übergeben, also genau der Wert, der beim Allokieren geliefert wurde. Mit „FreeAslRequest" können sowohl Strukturen freigegeben werden, die mit „AllocAslRequest" allokiert wurden, wie auch solche, die von „AllocFileRequest" stammen.

2.2.6 FreeFileRequest

„FreeFileRequest" gibt einen mit „AllocFileRequest" allokierten Requester frei.
FreeFileRequest(request)
A0

FreeFileRequest(struct FileRequest *)

Auch mit „FreeFileRequest" können Requester-Strukturen freigegeben werden. Und auch diese Funktion macht dabei keinen Unterschied, ob diese mit „AllocASLRequest" oder „AllocFileRequest" allokiert wurde. Sie ist, anders gesagt, völlig identisch zu „FreeAslRequest" und existiert nur, damit die Quelltexte besser aussehen.

2.3 Utility-Library

Die Utility-Library ist eine Sammlung von Funktionen, die irgendwie in keine andere Library gepaßt haben. Dementsprechend vielfältig ist auch ihr Anwendungsgebiet. Doch wenngleich man die meisten ihrer Funktionen ohne Probleme selbst programmieren könnte, so ist es doch bei weitem einfacher, auf eine Library zurückzugreifen. Man kann sich somit auf andere Probleme konzentrieren.

2.3.1 SMult32 / UMult32

„SMult32" multipliziert zwei 32-Bit Werte vorzeichenbehaftet, „UMult32" nicht vorzeichenbehaftet.

```
result = SMult32(arg1, arg2) /
         UMult32(arg1, arg2)
D0               D0   D1
```

LONG SMult32(LONG, LONG) / LONG UMult32(LONG, LONG)

„SMult32" und „UMult32" sind zwei Funktionen, die zwei gegebene 32-Bit Werte miteinander multiplizieren und als Ergebnis wieder einen 32-Bit Wert liefern. Da der 68000er nur 16-Bit Werte multiplizieren kann, und erst an höheren Prozessoren auch 32-Bit Werte verarbeitet werden können, sind diese beiden Funktionen der sichere

Weg, um sein Programm auf allen Amigas laufen zu lassen. „SMult32" berücksichtigt dabei das Vorzeichen, „UMult32" tut das nicht.

2.3.2 SDivMod32 / UDivMod32

„SDivMod32" dividiert 32-Bit Werte vorzeichenbehaftet, „UDivMod32" nicht vorzeichenbehaftet.

```
quotient:remainder = SDivMod32(DIVIDEND, DIVISOR) /
                     UDivMod32(DIVIDEND, DIVISOR)
D0    D1                D0    D1
```

LONG SDivMod32(LONG, LONG) / LONG UDivMod32(LONG, LONG)

„SDivMod32" und „UDivMod32" sind die 32-Bit Varianten der Division. Dabei werden alle Argumente sowie beide Resultate in 32-Bit gehandhabt. Es gibt zwei Resultate, da nicht nur das ganzzahlige Ergebnis der Division („quotient"), sondern auch der Rest („remainder") zurückgeliefert werden. Wieder liegt die Motivation in der Ermangelung einer 32-Bit Division des 68000ers. Auch dieses Funktionenpaar behandelt zum einen die vorzeichenbehaftete („SDivMod32"), zum anderen die nicht vorzeichenbehaftete („UDivMod32") Division. Als Argumente werden entsprechend Dividend und Divisor übergeben.

2.3.3 ToLower / ToUpper

„ToLower" konvertiert ein Zeichen in Kleinschrift, „ToUpper" in Großschrift.

```
char = ToLower(char)
       ToUpper(char)
D0             D0
```

CHAR ToLower(CHAR) / CHAR ToUpper(CHAR)

„ToLower" verwandelt ein gegebenes Zeichen in einen Kleinbuchstaben (wenn es das nicht bereits ist), „ToUpper" in einen Großbuchstaben. Das geht beim ASCII prinzipiell ja auch dadurch, daß man #$20 addiert bzw. subtrahiert, doch erwischt man mit dieser primitiven Methode nur die Zeichen A-Z und keine Umlaute. Und genau darauf zielen diese beiden Funktionen. Denn wenn die „locale.library" geladen ist, ersetzt diese die entsprechenden Funktionen der „utility.library" und behandelt dann auch alle Umlaute, und zwar abhängig von der gewählten Landessprache. Auf diese Weise kann man immer sicher sein, alle in Frage kommenden Buchstaben abgedeckt zu haben, auch wenn man mit den länderspezifischen Besonderheiten nicht vertraut ist.

Sowohl das Argument wie auch das Resultat sind Chars, also Zeichen. Es wäre zwar auch ganz nett, wenn man ganze Strings auf einmal behandeln könnte, aber so ist die Anwendung wenigstens flexibler, als wenn es nur die String-Variante gäbe.

2.3.4 Stricmp

„Stricmp" vergleicht zwei Strings ohne Rücksicht auf Groß-/ Kleinschreibung.
res = Stricmp(string1, string2)
D0 A0 A1

LONG Stricmp(char *, char*)

„Stricmp" vergleicht zwei Strings ohne dabei auf Groß-/ Kleinschreibung zu achten. Ähnlich wie bei „ToLower" / „ToUpper" wird auch diese Funktion bei geladener „locale.library" gegen eine entsprechende Funktion ersetzt, die länderspezifische Umlaute korrekt behandelt. Dies ist nicht nur in Hinsicht auf Groß-/ Kleinschreibung interessant, sondern auch ganz besonders beim Sortieren von Strings. Denn laut ASCII kommen beispielsweise die deutschen Umlaute ganz am Ende des Alphabets, in Wirklichkeit werden sie jedoch eingeordnet. Und genau dies wird von der „locale.library" dann unterstützt. Deshalb sollten alle Sortiervorgänge auf „Stricmp" aufbauen.

Als „res" wird entweder etwas negatives geliefert, wenn „string1" vor „string2" kommt, Null bei Gleichheit und etwas positives, wenn „string1" nach „string2" kommt. Sollten die beiden gegebenen Strings unterschiedliche Länge haben wird der kürzere so behandelt, als wäre er mit Nullen aufgefüllt.

2.3.5 Strnicmp

„Strnicmp" vergleicht zwei Strings ohne Rücksicht auf Groß-/ Kleinschreibung bis zu einer max. Länge.
res = Strnicmp(string1, string2, length)
D0 A0 A1 D0

LONG Strnicmp(char *, char *, LONG)

„Strnicmp" mach genau das gleiche wie „Stricmp", mit dem Unterschied, daß nur maximal „length" Zeichen miteinander verglichen werden.

2.3.6 Amiga2Date

„Amiga2Date" errechnet aus einem „timestamp" das tatsächliche Datum und die Zeit.
Amiga2Date(amigatime, date)
D0 A0

void Amiga2Date(ULONG, struct ClockData *)

Wenn es um die Handhabung von Datum und Zeit geht, ist der Amiga schon etwas seltsam. Praktisch überall trifft man auf den mysteriösen „timestamp", beispielsweise bei dem Erschaffungszeitpunkt von Files. Dieser Wert gibt die Anzahl der Sekunden an, die seit dem 1. Januar 1978 vergangen sind. Dies ist für das Betriebssystem

natürlich einfacher zu handhaben, als unser üblicher, sehr komplexes Format. Und da es viel öfter passiert, daß die Zeit sich verändert (nämlich jede Sekunde) als daß sie abgefragt wird, ist dies auch ein sinnvolles Vorgehen. Das Problem liegt nur darin, daß wenn nun einmal Datum und/oder Zeit abgefragt werden müssen, der tatsächliche Wert nur schwer zu errechnen ist. Man bedenke nur die Schaltjahre.
Um dem Programmierer die Erstellung einer Routine, die dieses Problem fehlerfrei löst, zu ersparen, gibt es nun drei Funktionen, die sich mit der Amiga-Zeitrechnung befassen. Die erste ist „Amiga2Date" (sprich: Amiga to date). Sie nimmt den „timestamp" als „amigatime" und eine „ClockData"-Struktur als „date" entgegen und füllt diese Struktur dann mit dem korrekten Werten. Da die Struktur, die das Ergebnis letztendlich enthält bereits beim Aufruf mit übergeben wird, werden auch keine weiteren Resultate zurückgeliefert.

2.3.7 Date2Amiga

„Date2Amiga" errchnet aus dem tatsächlichen Datum und der Zeit einen „timestamp".
amigatime = Date2Amiga(date)
D0 A0

ULONG Date2Amiga(struct ClockData *)

„Date2Amiga" geht den umgekehrten Weg: Diese Funktion nimmt eine „ClockData"-Struktur entgegen und berechnet aus ihr den „timestamp". Dieser Wert wird anschließend als „amigatime" zurückgeliefert. Beachten Sie dabei, daß das übergebene Datum nicht auf Gültigkeit geprüft wird. Sie sollten also sicherstellen, daß nur vernünftige

Werte eingesetzt werden. Zum Überprüfen eignet sich die folgende Funktion:

2.3.8 CheckDate

„CheckDate" überprüft eine „ClockData"-Struktur auf ein gültiges Datum.

```
amigatime = CheckDate(Date)
D0                    A0
```

ULONG CheckDate(struct ClockData *)

„CheckDate" errechnet, ebenso wie „Date2Amiga" den „timestamp" aus einer gegebenen „ClockData"-Struktur. Allerdings wird bei dieser Funktion die Gültigkeit der angegebenen Werte überprüft und, sollte sich herausstellen, daß das Datum nicht existiert, eine Null zurückgegeben. Im allgemeinen ist es daher besser, Umrechnungen mit „CheckDate" vornehmen zu lassen und den Rückgabewert zu überprüfen.

Nachwort

Sie haben es also geschafft! Nachdem Sie nun (hoffentlich) den Blitzeinstieg komplett durchgelesen haben, kann ich Ihnen nur gratulieren: Jetzt wissen Sie, was Sache ist. Nichts ist auf einem Computer so wichtig, wie grundlegendes Wissen. Es hilft Ihnen nichts, wenn Sie hundert verschiedene Programme perfekt bedienen können, und wenn Sie sich an ein neues setzen, haben Sie keine Ahnung wo Sie anfangen sollen. Ich hoffe, ich konnte Ihnen etwas von dem prinzipiellen Vorgehen beibringen, wenngleich Sie dabei auch alle Workbench-Programme kennengelernt haben. Aber es kommt ja gar nicht darauf an, daß Sie das „ScreenMode"-Programm im Schlaf bedienen können; wenn Ihnen eine Funktion entfällt, wissen Sie ja, wo Sie nachschlagen können. Die Hauptsache ist, daß Sie ein Verständnis für AmigaOS 2.0 entwickelt haben, mit den Gadgets zurechtkommen und den Aufbau der Workbench verstehen. Und auch der Shell sollten Sie sich nun ohne Angst und Schrecken nähern können - bedenken Sie immer, daß Generationen von PC-Anwendern mit MS-DOS eine viel unangenehmere Umgebung hatten.
Warten wir also gemeinsam auf AmigaOS 3.0 und hoffen, daß es noch viel mehr Neuerungen mit sich bringt, die das Leben mit dem Amiga noch einfacher machen!
Peter Eberlein